ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА
(повне найменування вищого навчального закладу)

Факультет математики, фізики та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра математичного забезпечення комп'ютерних систем
(повна назва кафедри (предметної, циклової комісії))

Кваліфікаційна робота

на здобуття ступеня вищої освіти « бакалавр »

«Розробка системи ведення статистики відвідувань веб-сайтів»

«Development of a system for keeping statistics of website visits»

Виконав: студент денної форми навчання
спеціальності 123 – Комп'ютерна інженерія.
(шифр і назва напряму підготовки, спеціальності)
Освітня програма «Комп'ютерна інженерія»
(назва освітньої програми)
Осіпова Ольга Сергіївна
(прізвище, ім'я, по-батькові)

Керівник доцент, док.тех.наук Михайленко В.С.
(науковий ступінь, вчене звання, прізвище та ініціали, **підпис**)

Рецензент старший викладач Шаріпова І.В.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
№___ від «___»_____ 2024 р.

Завідувач кафедри
_____ _Юрій Гунченко_
(підпис) (ім'я, прізвище)

Захищено на засіданні ЕК № ___
протокол № __ від «___» ____ 2024 р.
Оцінка _____/_____/_____
(за національною шкалою, шкалою ECTS, бали)
Голова ЕК
_____ _ _
(підпис) (ім'я, прізвище)

Одеса - 2024

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА

(повне найменування вищого навчального закладу)

Факультет ___математики, фізики та інформаційних технологій_____

Кафедра ____комп'ютерних систем та технологій_____

Освітньо-кваліфікаційний рівень_____бакалавр_____

Спеціальність _____123 – Комп'ютерна інженерія_____

(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____

проф. Юрій ГУНЧЕНКО__

"____" _____2024 року

З А В Д А Н Н Я
на **кваліфікаційну роботу студенту**

Осіпова Ольга Сергіївна

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Розробка системи ведення статистики відвідувань веб-сайтів
_____
_____

керівник кваліфікаційної роботи
д.т.н. доцент Михайленко В.С._____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 18 червня 2024 р._____

3. Вихідні дані роботи
Літературні джерела відповідні до тематики роботи_____
_____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Вступ, 1 Теоретична частина, 1.1 Значення пошукових систем, 1.2 Digital маркетинг та SEO, 1.3 Основні елементи веб-аналітики, 2 Вибір програмного забезпечення, 3 Практична реалізація, 3.1 Аналіз систем відвідування веб-сайтів, 3.1.1 Загальний принцип побудови лічильників відвідувань веб-сторінок, 3.1.2 Лічильник з використанням графіки, 3.1.3 Лічильник у вигляді вбудованого модулю у сайт, 3.1.4 Комбінований лічильник, 3.2 Створення програмного застосунку, 3.3 Аналіз технічного завдання, 3.4 Створення бази даних, 3.5 Розробка інформаційної системи, 3.6 Принцип роботу модулю для збору статистики, 3.7 Встановлення модулю для збору статистики, 3.7.1 Помилки з якими можна зіткнутись, 4 Інструкція користувача, 5 Проведення аналітичних тестів, Висновок, Список використаних джерел, Додаток А, Додаток Б.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
1. Актуальність теми, 2. Мета та Постановка задачі, 3. Популярні веб-браузери, 4. Значущість SEO, 5. Маркетингові комунікації, 6. Популярні сервіси для аналітики, 7. Переваги програми, 8. Засоби реалізації, 9. Перегляд застосунку, 10. Висновок

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
5	д.т.н. доцент Михайленко В.С.		

7. Дата видачі завдання 25 січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Ознайомлення з завданням до кваліфікаційної роботи	25.01.2024	*Виконано*
2	Підбір джерел по темі роботи	26.01.24 – 15.03.24	*Виконано*
3	Оформлення першого розділу	19.04.2024	*Виконано*
4	Оформлення другого розділу	27.04.2024	*Виконано*
5	Оформлення третього розділу	30.04.2024	*Виконано*
6	Оформлення четвертого розділу	15.05.2024	*Виконано*
7	Оформлення п’ятого розділу	20.05.2024	*Виконано*
8	Виконала завдання до розділу «Проведення аналітичних тестів»	23.05.2024	*Виконано*
9	Оформлення кваліфікаційної роботи	02.06.2024	*Виконано*

Студент _____ _____
(ім’я, прізвище)

Керівник проекту (роботи) _____ _____
(ім’я, прізвище)

АНОТАЦІЯ

Тема роботи: Розробка компоненту веб-сайту для ведення статистики відвідувань.

Актуальність теми. У сучасному світі, де інформаційні технології та інтернет стали невід'ємною частиною життя більшості людей, важливою задачею для власників веб-ресурсів є збір та аналіз статистики відвідувань. Це дозволяє не тільки покращувати контент та інтерфейс, але й оптимізувати маркетингові та рекламні стратегії. Розробка надійного, гнучкого інструменту для аналізу поведінки користувачів є критично важливою для підвищення ефективності веб-сайтів.

Мета роботи полягає у створенні модуля для веб-сайтів, який здійснює збір статистичних даних про відвідування: кількість користувачів, їхні демографічні характеристики (країна проживання), а також технічні параметри (типи пристроїв, веб-браузер). Зібрана інформація аналізується і використовується для адаптації контенту, структури, дизайну сайту, а також для формулювання стратегій щодо його просування. Об'єкт дослідження - процеси збору та аналізу статистичних даних відвідувачів веб-сайтів. Предмет дослідження - компонент веб-сайту для ведення статистики відвідувань.

Методи дослідження. У роботі використовувались методи аналізу предметної області, програмування, баз даних та веб-дизайну, що включали: аналіз предметної області для визначення основних вимог та функціоналу статистичного модуля, вивчення існуючих аналогів для аналізу конкурентів та визначення недоліків та переваг їхніх продуктів, програмування веб-застосунку для реалізації створення, збору, та аналізу даних, створення бази даних для ефективного зберігання та обробки зібраних статистичних даних.

ANNOTATION

Topic: Development of a website component for tracking visitor statistics.

Relevance. In today's world, where information technology and the internet have become an integral part of most people's lives, an important task for website owners is collecting and analyzing visitor statistics. This not only allows for content and interface improvement but also optimizes marketing and advertising strategies. Developing a reliable, flexible tool for analyzing user behavior is critically important for enhancing website effectiveness.

The purpose of this work is to create a module for websites that collects statistical data on visits: the number of users, their demographic characteristics, preferences, and technical parameters (device types, screen resolutions). The collected information is analyzed and used to adapt the site's content, structure, design, and to formulate promotion strategies. The object of the research is the processes of collecting and analyzing statistical data of website visitors. The subject of the research is the website component for tracking visitor statistics.

Research methods. The study utilized methods of domain analysis, programming, databases, and web design, which included: domain analysis to determine the main requirements and functionality of the statistical module, studying existing analogs to analyze competitors and identify the advantages and disadvantages of their products, web application programming for implementing data collection and analysis, database creation for efficient storage and processing of collected statistical data.

ЗМІСТ

ВСТУП

В епоху розвитку інформаційних технологій, а також ріст використання мережі інтернет серед всіх слоїв населення, збільшило попит на різні мережеві ресурси, такі як: інтернет-магазини, мобільні додатки, інструменти для веб-аналітики.

Даний дипломний проект спрямований на проектування, аналіз, а також розробку компоненту сайту для ведення статистики відвідувань. Цей інструмент дозволить власнику сайту отримувати інформацію про кількість користувачів, їх вік, стать, вподобання, які відвідували його ресурс за певний пробіжок часу. Також можливість побудови гістограми, яка буде відображати статистику відвідувань.

Метою даної роботи є створення додаткового модуля, який збиратиме інформацію про використання ресурсів сайту відвідувачами в базу даних та допоможе адаптувати її до структури сайту, дизайну, контенту та технічної реалізації функціоналу на аналізі цих даних. Це розмір сторінки, використання різноманітних мультимедійних елементів, увага до роздільної здатності моніторів відвідувачів вашого сайту тощо. За допомогою такого модуля збору статистичних даних, вбудованого в сторінку веб-сайту, ця інформація збиратиметься та фіксуватиметься в базі даних, а потім використовуватиметься для формулювання цільових заходів для просування мережевих ресурсів у потрібному напрямку. Для досягнення зазначеної мети необхідно виконати наступні задачі:

1) Аналіз предметної області;
2) Аналіз конкурентів, виявлення достатків та недоліків їх продуктів;
3) Програмування застосунку (створення веб-сайту);
4) Створення бази даних

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Значення пошукових систем

Пошукові системи – це певні бази даних – онлайн служби, представлені апаратно-програмним комплексом з наявним web-інтерфейсом, які відкривають доступ пошуку будь-якої інформації в мережі інтернет [1].

Пошукові системи використовують Інтернет для просування бізнесу та допомагають користувачам знаходити вміст, пов'язаний із конкретними запитами, серед незліченної кількості веб-сайтів.

Хронологія появи сучасних пошукових систем починається з прототипу сервісу Archie. Свого роду онлайн-каталогу, де ви можете шукати потрібну вам інформацію за назвою.

У 1993 році користувачам Інтернету була представлена проста пошукова система під назвою W3Catalog. У 1995 році-Yahoo. У 1997 році Google впевнено заявив, що це система обману, створена як навчальний проект для студентів Стенфорда, які створили пошукову систему BackRub у 1996 році. Вона лягла в основу пошукової системи Google. Поступово пошукові системи стали не тільки важливими інструментами користувачів. Сьогодні це ефективні обов’язкові майданчики для просування бізнесу.

Пошукові системи постійно аналізують терабайти даних, розміщених на мільярдах веб-сторінок, і зосереджуються на сайтах, пов'язаних із запитами користувачів, але решта просто ігнорується.

Сьогодні пошукові системи є більш широким типом систем, заснованих на класичних нелінійних алгоритмах, і роботи здатні складати найточніший рейтинг веб-сайтів за сотнями критеріїв. Це означає, що кожен бренд або проект потребує розробки професійного веб-сайту. Якісне створення, налаштування та підтримка необхідних параметрів дозволяє домогтися високого рівня довіри до пошукових

роботів, необхідного для хорошої відвідуваності і конверсії з боку потенційних клієнтів.

Кожна пошукова система має спеціальні критерії для алгоритмів пошуку даних. Але ранжування результатів і принцип роботи пошукових роботів є загальними. Процес пошуку інформації в мережі:

– збір даних з інтернет-сайтів;

– індексація web сайтів;

– пошук за запитами;

– ранжування результатів видачі.

Індексація веб-сторінок

Декомунізація – це початок SEO, тобто пошукової оптимізації. Декомбінація – це процес сканування сайту пошуковим роботом, під час якого сторінка додається до відповідної бази даних. Це необхідно для декомунізації доступу користувача до пошукової системи за допомогою відповідних запитів.

Сторінки сайту включені в бази даних Google, Bing та інших пошукових систем. Користувачі побачать це, і це буде шансом для компаній залучити нових клієнтів.

Алгоритм сортування

Алгоритм сортування – це набір правил для автоматичного створення результатів, найближчих до надісланого запиту. Широкий спектр алгоритмів і факторів ранжування постійно змінюється і вдосконалюється. Щоб SEO-сайт залишався у верхній частині результатів пошуку, важливо враховувати алгоритм ранжування в процесі декомунізації SEO-сайту.

Персоналізація результатів пошуку

Опція персоналізації дозволяє користувачу отримувати результати пошуку Google, які формуються відповідно до його дій, інтересів [1].

Найпопулярніші пошукові системи

В Україні майже 98% користувачів віддають перевагу пошуковій системі Google, то в усьому світі, особливо в США, вибір набагато ширший. Це пошукові системи, які можливо використовувались, чи ті, про які можна почути вперше. Залежно від популярності використання слід виділити наступні пошукові системи, серед яких відомі:

– Google – понад 85% користувачів.

– Bing – біля 7% відвідувачів.

– Yahoo!

– DuckDuckGo

– Baidu .

Діаграма популярності використання пошукових систем в Україні (див. рис. 1.1).

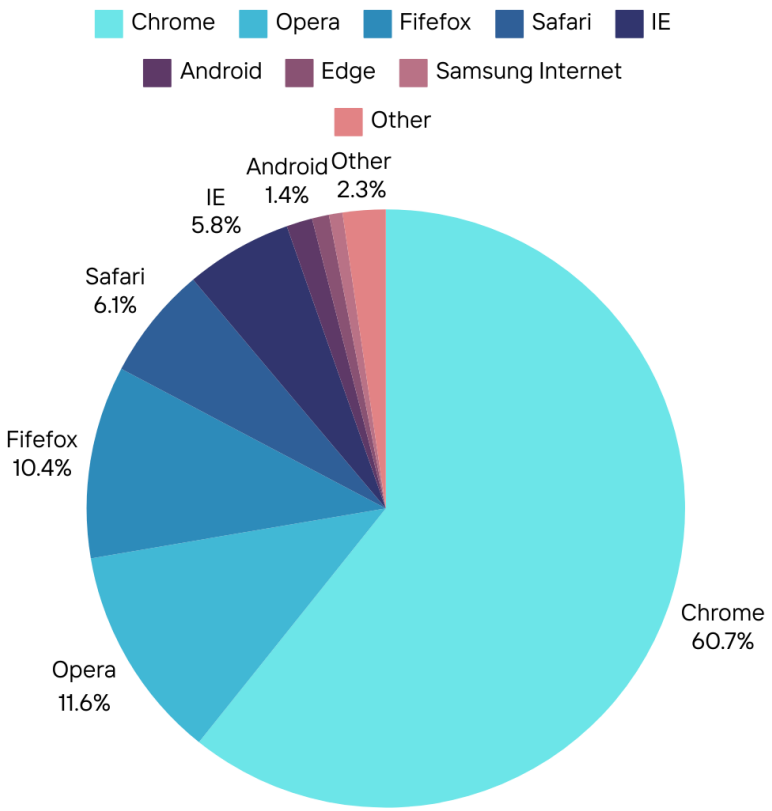

Рисунок 1.1 – Діаграма популярності пошукових систем

1.2 Digital маркетинг та SEO

Зв'язок між SEO і цифровим маркет декомунізацією може мати вирішальне значення для вашої компанії в цифровому просторі. Оптимізація веб - сайту для пошукових систем-це лише одна частина великого ланцюжка, яка допомагає залучати нових клієнтів і збільшувати доходи.

За допомогою цифрових медіа-каналів, таких як веб-сайти, соціальні мережі, електронна пошта, пошукові деки, мобільні додатки та інші технології, пов'язані з Інтернетом, можна залучити більше клієнтів, розширити базу, підвищити впізнаваність бренду і збільшити продажі. Це також забезпечує більш ефективне спілкування для досягнення ваших маркетингових цілей.

Маркетингова комунікація-це процес створення та розповсюдження повідомлень про товари та послуги з метою привернення уваги потенційних клієнтів та збільшення продажів. Цей процес включає різні види спілкування, включаючи рекламу, PR, особисті продажі, прямий маркетинг, спонсорство та упаковку товару. Основною метою маркетингових комунікацій є привернення уваги, зацікавленість цільової аудиторії, створення позитивного іміджу бренду і сприяння розвитку продуктів і послуг [2].

Важливо пам’ятати, що кожен бізнес має свої унікальні атрибути та вимоги, тому рекомендації можуть відрізнятися для різних типів бізнесу та цільової аудиторії.

Існує кілька методів збільшення прибутку за допомогою веб-сайту.

− Оптимізуйте свій веб-сайт для пошукових систем (SEO) - це підвищить видимість вашого веб-сайту в пошукових системах, що призведе до збільшення відвідувачів і потенційних клієнтів.

− Надайте простий, зрозумілий інтерфейс користувача, який зменшить кількість відвідувачів, які залишають сайт через плутанину або труднощі з використанням функцій.

− Використовуйте візуальний вміст – зображення, відео та інші засоби масової інформації можуть залучати відвідувачів і рекламувати продукти.

− Збільште довіру клієнтів. Надайте конкретну інформацію про послуги та продукти, публікуйте відгуки клієнтів, використовуйте гарантії та сертифікати, щоб підвищити довіру та збільшити дохід.

Пошукова оптимізація (англ. SEO) — це оптимізація сайту для подальшого його просування в рейтингу пошукових систем. Вона допомагає підвищити органічний трафік на сайт компанії, збільшити ROI. Пошукова оптимізація дає можливість бренду завоювати вищий рівень довіри, донести повідомлення до цільової аудиторії та покращити досвід покупця [3].

Пошукова оптимізація стане ефективним інструментом залучення уваги до сайту, збільшення органічного трафіку і підвищення надійності бренду на ринку. SEO може допомогти залучити більше потенційних клієнтів, просувати свій вміст, заохочувати потенційних клієнтів купувати та надавати їм конкурентні переваги. Крім того, SEO - це ефективний спосіб зайняти позицію вище результатів пошуку в пошукових системах. Таким чином, сайт стане більш помітним для клієнтів. Як правило, користувачі переходять на сайт з найвищими результатами в Google.

SEO-оптимізація-це безперервний процес, який вимагає пильної уваги до деталей. Але якщо у вас є стратегія, seo-оптимізація може допомогти вам збільшити відвідуваність вашого сайту і збільшити конверсії. Це допоможе забезпечити стабільність вашого бізнесу. Важливо розуміти, що алгоритми постійно змінюються, тому ваша SEO-стратегія повинна постійно оновлюватися і адаптуватися до вимог пошукової системи.

Рекомендації щодо проведення дії для SEO оптимізації сайту:

− Ключові слова: почніть з визначення ключових слів, які відображають вашу тему. Ви можете використовувати такі інструменти, як Google Keyword Planner, Ahrefs та SEMrush, щоб знайти потрібні ключові слова.

− Оптимізація вмісту: створюючи вміст для статей, блогів, веб-сторінок тощо, дотримуйтесь правил SEO. Наприклад, використовуйте ключові слова в заголовках, описах, мета-тегах та вмісті, але не зловживайте ними.

− Забезпечте швидке завантаження веб-сторінок: швидкість завантаження веб-сторінок важлива для SEO. Ви можете використовувати кешування, мінімізувати кількість файлів CSS та js, а також використовувати Cdn (мережі доставки вмісту), щоб скоротити час завантаження веб-сторінок.

− Забезпечте хорошу взаємодію з користувачем: забезпечте хорошу взаємодію з користувачем, особливо щодо простоти навігації, ясності та доступності вмісту.

− Зворотні посилання: залучення зворотних посилань - це процес отримання гіперпосилань на ваш сайт з інших веб-сторінок. Вони допомагають підвищити надійність Вашого сайту і його рейтинг в пошукових системах.

− Моніторинг: відстежуйте свої SEO-кампанії та результати. Використовуйте такі інструменти, як Google Analytics і Google Search Console, для аналізу відвідуваності вашого сайту і коригування вашої SEO-стратегії [2].

1.3 Основні елементи веб-аналітики

Веб-аналітика є способом збору, вимірювання, аналізу даних про всі відвідувачів сайтів і призначена для оптимізації їх роботи. На підставі отриманої інформації можна скласти план по збільшенню і розвитку можливостей ресурсу. Отримана статистика допомагає вивчити кількість переглянутих відвідувачем сторінок, географію аудиторії, ключові фрази для пошукових систем, цільову аудиторію і випадкових відвідувачів, а також зручність навігації і користування сайтом.

Для правильного відображення даних застосовується відразу кілька інструментів, які забезпечують точність отриманої інформації. Всього існує два основних види інструментів:

– лічильники – на сайт встановлюється фрагмент коду, дані збираються в одну базу, перевагою є простота і зручність використання, оперативне отримання наочної інформації;

– лог-аналізатори – встановлюються на ПК користувача, дані зберігаються в архіві, за допомогою них можна отримувати точнішу інформацію, вирішувати складні завдання, аналізувати помилки і збирати статистику по трафіку ресурсу.

Вам потрібно знати контейнер Google Tag Manager, а потім почати використовувати лічильник. За допомогою програмістів достатньо буде один раз встановити контейнер в код сайту, а потім самостійно встановлювати всі наступні лічильники.

Найбільш популярними лічильниками є GoogleAnalytics, Openstat і Yandex.Metrica та найпоширеніші аналізатори журналів-це Webalizer, Awstats та інші. Нижче представлена таблиця порівняння вже існуючих та досить популярних лічильників.

Таблиця 1.1 Порівняння найпопулярніших лічильників

Параметр	Google Analytics	Openstat	Webalizer	AWStats
Тип	Лічильник	Лічильник	Лог-аналізатор	Лог-аналізатор
Власник	Google	Openstat	Незалежний	Незалежний
Вартість	Безкоштовно / Платні функції	Безкоштовно	Безкоштовно	Безкоштовно
Метод збору даних	JavaScript теги, API	JavaScript теги	Аналіз лог-файлів	Аналіз лог-файлів

Продовження таблиці 1.1

Глибина аналітики	Висока	Середня	Низька	Середня
Інтеграція з іншими сервісами	Так (Google Ads, Search Console та інші)	Обмежена	Ні	Ні
Мобільний доступ	Так	Так	Ні	Ні
Доступ до сирих даних	Обмежений (тільки в платних версіях)	Немає	Так	Так
Простота налаштування	Відносно складне	Просте	Відносно складне	Відносно складне
Інтерфейс	Веб-інтерфейс	Веб-інтерфейс	HTML-звіти	HTML-звіти
Реальний час	Так	Так	Ні	Ні
Події та цілі	Так	Так	Ні	Ні
Підтримка історичних даних	Так	Так	Так	Так

GoogleAnalytics залишається класичним інструментом для всіх веб-аналітиків. Це абсолютно безкоштовний сервіс, який надає користувачам можливість створювати детальну статистику відвідувачів сайту. Користувач встановлює на сторінці свого ресурсу JavaScript-код, а всі дані збираються на

сервері Google. Програма спрацьовує під час відкриття відвідувачем сторінки в браузері на своєму комп'ютері. За допомогою цього сервісу можна отримати різні показники:

– кількість переглянутих сторінок;

– деталізований шлях відвідувача на сайті;

– загальна кількість відвідувачів;

– ресурси, з яких прийшли користувачі;

– пошукові запити, за якими перейшли клієнти;

– визначається за IP-адресою географічне положення користувачів;

– кількість часу, проведений відвідувачем на сайті.

Завдяки цьому можна отримати інформацію про зацікавленість цільової аудиторії і підвищити якість сайту.

Яндекс.Метрика є аналогом GoogleAnalytics, діє за тим же принципом і використовується для аналізу дій відвідувачів і оцінки трафіку. Інформація надається за поточний день і оновлюється з періодичністю раз в п'ять хвилин. Основним звітом є звіт конверсії, який показує кількість візитів цільової аудиторії, замовлень, ступінь конверсії, загальні доходи ресурсу і коефіцієнт виконання поставленої мети. Ще одним важливим звітом є вікова структура, що дозволяє порівняти демографічні параметри користувачів.

Webalizer – це лог-аналізатор, прикладна програма, яка надає щоденні та щомісячні звіти про відвідування сайту. Програма дозволяє записувати дії відвідувачів у спеціальні файли, статистику збирає сервер хостинг-провайдера. Вона генерує HTML-сторінки про роботу сайту на основі файлів реєстрації користувачів. Інструмент дозволяє створювати аналітику кількості відвідувань, вихідних сторінок, розташування відвідувачів, кількість завантаженої інформації і багато іншого [4].

2 ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для створення програми обрали Opencart, який буде виступати у ролі макету. За допомогою цього шаблону, побудуємо основу для майбутнього сайту з статистикою відвідувань.

Opencart має чудові можливості як і для реалізації сторінок які бачать клієнт коли заходить на сайт, а також застосунок для адміністраторів, які можуть управляти сайтом, а також продивлятись статистику (показано на рис. 2.1).

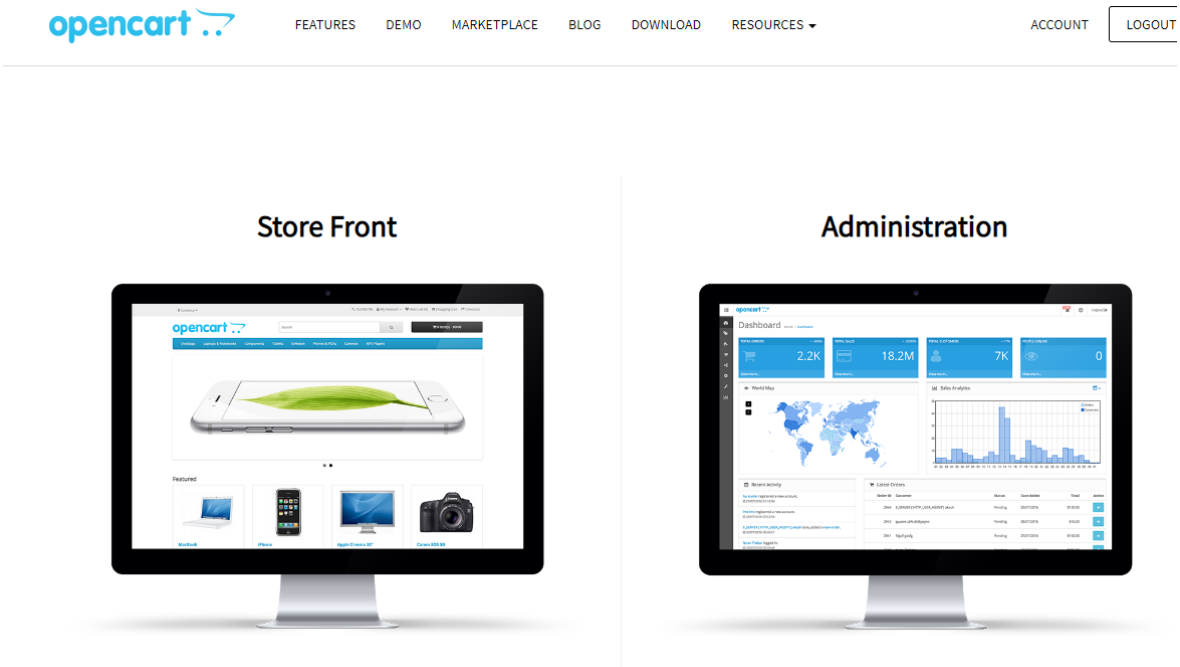

Рисунок 2.1 – Зовнішній вигляд застосунку Opencart

Так як Opencart створений на мові програмування PHP, то для роботи з ним використали XAMPP. Ця програма дає можливість підключати та працювати з базою даних, а також веб-сервером через панель контролювання (показано на рис. 2.2). XAMPP - це безкоштовний стек рішень для веб-сервера з відкритим вихідним кодом. Це допомагає розробникам тестувати свої веб-

додатки або веб-сайти у своєму середовищі розробки. У нього є всі необхідні компоненти, включно з Apache, Perl, MySQL Database і PHP. Xampp надає надійне середовище розробки локального веб-сервера для веб-додатків на основі PERL і PHP. Крім того, він надає MariaDB і MySQL для управління базами даних. Після успішної інсталяції XAMPP ви можете запустити і зупинити кожен модуль, використовуючи панель управління XAMPP. Для тестування додатків PHP вам потрібно лише запустити два модулі Apache і MySQL. Це дозволить PHP- програмам працювати на вашому комп'ютері.

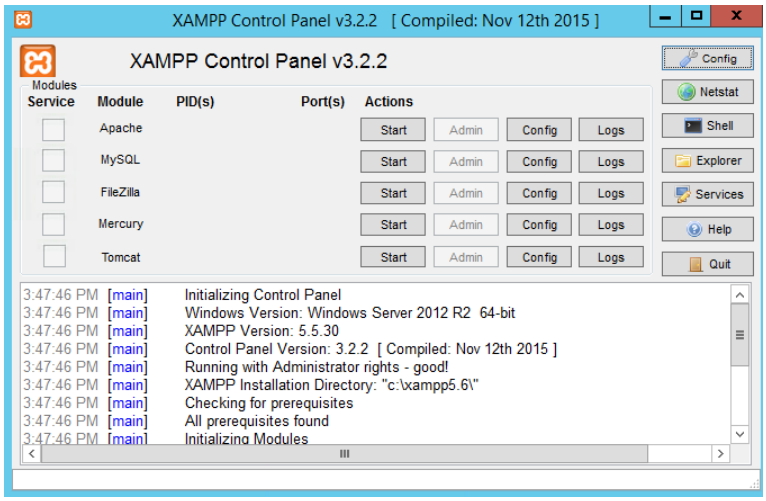

Рисунок 2.2 – Панель керування XAMPP

Обираючи СУБД, для реалізації рівня управління ресурсами інформаційної системи, обрали MySQL за такими причинами:

1) Швидкість та продуктивність: MySQL відомий своєю високою продуктивністю та швидкістю обробки запитів, що робить його ідеальним для програм, які потребують швидкого доступу до даних.

2) Безпека: MySQL пропонує розширені функції безпеки, такі як шифрування даних, управління користувачами та права доступу, щоб забезпечити високий рівень захисту даних.

3) Широка спільнота та підтримка: MySQL має велику спільноту користувачів та розробників, які надають доступ до широкого спектру ресурсів, документації та підтримки.

4) Сумісність та стандартна підтримка: MySQL підтримує стандартний sql для забезпечення сумісності з різними програмами та інструментами. Він також легко інтегрується з багатьма мовами програмування та середовищами розробки.

5) Рішення з відкритим кодом та комерційні рішення: MySQL доступний як у відкритих, так і в комерційних версіях, і ви можете вибрати декомунізовані безкоштовні та платні варіанти з розширеними функціями та підтримкою.

6) Резервне копіювання та відновлення: MySQL надає різні механізми резервного копіювання та відновлення ваших даних, це дуже важливо для зберігання і безпеки інформації.

Використали HTML і CSS разом з JavaScript для створення веб-додатку інформаційної системи:

– HTML (Hypertext Markup Language), також відома в IT сфері як мова розмітки гіпертекстових документів, по факту є набором спеціальних елементів - тегів. Вони допомагають структурувати кожну сторінку Web сайту.

– CSS (Cascading Style Sheets - таблиця каскадних стилів) - мова за допомогою якої оформлюються усі веб сторінки. Для того, щоб підключити CSS стиль до певного HTML тегу чи сукупності тегів використовуються спеціальні HTML атрибути class чи id. Наприклад, class="style_page" або id="torro-2".

– HTML і CSS разом з JavaScript утворюють основу сучасного веб-дизайну. З їх допомогою будь яка інформація, яку містить Веб сторінка візуалізується, формується, стилізується та упорядковується [5].

Обирали Apache для серверної частини з наступних причин:

1) Надійність та стабільність: Apache відомий своєю надійністю та стабільністю, що робить його одним із найпопулярніших веб-серверів у світі.1 він може задовольнити попит на велику кількість без великих затримок.

2) Гнучкість і розширюваність: Apache пропонує безліч модулів, які дозволяють розширити його функціональність. Ви можете налаштувати сервер для підтримки різних мов програмування, баз даних та інших технологій.

3) Безпека: Apache має розширені функції безпеки, такі як шифрування SSL / TLS, контроль доступу та можливість встановлювати різні політики безпеки за допомогою файлів конфігурації.

4) Кросплатформний: працює на декількох операційних системах, включаючи Apache, Unix, Linux, Windows і деку macOS.Це дає вам велику гнучкість у виборі правильної платформи для вашого сервера.

5) Широка спільнота та підтримка: Apache має велику спільноту користувачів та розробників, які надають доступ до широкого спектру ресурсів, документації та підтримки. Це полегшить вирішення проблеми та пошук рішення для конкретного завдання.

6) Ліцензія: Apache-це програмне забезпечення з відкритим кодом, яке розповсюджується відповідно до Apache License2.0. Це дозволяє використовувати його безкоштовно як для особистих, так і для комерційних проектів.

Обираючи Node.js для серверної частини курсового проекту з інформаційної системи для модельного агентства, обрали з наступних причин.

Node.js - це однопотокове кросплатформенне середовище виконання з відкритим вихідним кодом і бібліотека, що використовується для запуску веб-додатків, написаних на JavaScript, поза браузером клієнта. Node.js - це програмне середовище, яке дає змогу запускати програми, написані мовою Javascript, поза браузером.

Під час розробки Node.js за основу було взято рушій виконання JavaScript під назвою V8, який був створений компанією Google і використовувався в браузері Google Chrome. Оскільки після створення Node.js Javascript код можна запустити фактично в будь-якому середовищі, за допомогою цієї бібліотеки можна написати не тільки фронтенд, а й серверну частину веб-додатка. Простіше кажучи, це означає, що цілі сайти тепер можуть працювати з використанням єдиного «стека», що робить розробку та обслуговування набагато швидшим і легшим, дозволяючи зосередитися на досягненні бізнес-цілей проекту [6].

Обираючи фрейворк React.js, опиралися на такі чинники. Фреймворк React.js - це фреймворк і бібліотека JavaScript з відкритим вихідним кодом, розроблені та підтримувані Facebook та Instagram. React розробляє програми та проекти, створюючи повторно використовувані компоненти, які можна розглядати як незалежні блоки Lego, використовуючи значно менше коду, ніж звичайний JavaScript, для швидкого та ефективного створення інтерактивних інтерфейсів користувача та веб-додатків. Ці компоненти є окремими частинами кінцевого інтерфейсу і, будучи зібраними, утворюють весь користувальницький інтерфейс програми. Основна роль React у застосунку - управління відображенням цього застосунку так само, як буква V у шаблоні «модель-представлення-контролер» (MVC), забезпечуючи оптимальне та найефективніше виконання рендерингу. Замість того, щоб розглядати весь інтерфейс користувача як єдине ціле, React.JS пропонується розробникам розділити ці складні інтерфейси користувача на окремі повторно використовувані компоненти, які утворюють будівельні блоки всього інтерфейсу користувача [7].

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Аналіз систем відвідування веб-сайтів

Мета цього розділу-пояснити різні типи лічильників сеансів та основні принципи їх роботи. Це допоможе вибрати правильний тип лічильника перед проектуванням системи.

Аналіз статистики сайту заснований на даних про відвідувачів сайту. Дані про відвідувачів збираються веб-сервером (і записуються в лог-файли) або лічильником сеансів (і заносяться в базу даних).

3.1.1 Загальний принцип побудови лічильників відвідувань веб-сторінок

Принцип роботи всіх лічильників полягає в запуску зовнішніх програм при завантаженні сторінок сайту. При завантаженні лічильника виконується зовнішня програма і в лічильник передаються так звані змінні оточення. У цих змінних зберігається вся основна інформація про поточного відвідувача сайту:

1) IP-адреса відвідувача (REMOTE_ADDR);

2) Браузер відвідувача (HTTP_USER_AGENT);

3) Адреса сторінки, на яку перейшов відвідувач (HTTP_REFERER);

4) Адреса сторінки, з якої перейшов відвідувач (REQUEST_URI);

5) Параметр виклику сторінки (QUERY_STRING).

Параметри виклику або QUERY_STRING передаються через знак питання ? після адреси сторінки і розділяються знаком амперсанда &, наприклад:

```
//cgi/test.php?i=34433&resolution=2048&color=64
```

Виклик зовнішньої програми може бути виконаний різними способами, але найбільш поширеним є використання зображення.

3.1.2 Лічильник з використанням графіки

Сторінки веб-сайту зазвичай складаються з тексту та графіки. Текст форматується за допомогою спеціальних тегів, а графіка являє собою набір

зображень, розміщених в потрібному місці.

Зображення вставляється на сторінку, як показано в лістингу 3.1:

Лістинг 3.1 – Вставка зображення-лічильника у HTML-код

```
... text of html page..
<img          src=http://www.myserver.com/img/picture.gif
              width=720 height=48>
... text of html page..
```

Браузер відображає сторінку на екрані, `http://www.myserver.com/img/picture.gif`, а у відповідь сервер надсилає файл зображення до браузера.

Принцип роботи лічильника зображень передбачає, що при перегляді сайту браузер користувача автоматично завантажує всі зображення і відповідно завантажує зображення лічильника. Ось 2 основні проблеми:

– Деякі користувачі відключають завантаження зображень;

– Робот не завантажує всі зображення під час сканування веб-сайту.

Таким чином неможливо підрахувати користувачів, у яких вимкнено завантаження зображень, і зникає можливість контролювати активність робота на сайті.

Як розраховується статистика на основі зображень лічильника.

Замість посилання на зображення вставляється виклик зовнішньої програми, яка має уявний вид зображення:

```
<img src=http://www.myserver.com/counter.php height=0 width=0>
```

Програма counter.php генерує файл зображення, який передається браузеру. Отже, у випадку з браузером, коли викликається програму, здається, що вона завантажує звичайне зображення.

Зображення, яке генерує програма, може бути будь-яким. Наприклад, прозорий GIF-файл розміром 1x1 або зображення з лічильником 88x31, яке

містить кількість відвідувань сайту (загальна кількість переглядів сайту, сьогоднішній перегляд, сьогоднішні користувачі).

Аналізуючи змінні середовища, програма отримує IP-адресу відвідувача і дані браузера, записує цю інформацію в базу даних для подальшого аналізу. Однак для отримання повноцінної статистики потрібна додаткова інформація. Для такого випадку, в програмі можна використовувати рядок параметрів виклику. Наприклад, передача роздільної здатності екрану користувача спрощено виглядає так:

```
<img src=http://www.myserver.com/counter.php?screen=2048
height=0 width=0>
```

Додаткова інформація про користувача виходить за допомогою java-скриптів: посилання, роздільна здатність єкрану, глибина кольору, випадкове число, інформація про встановлені файли cookie і т.д. Тому, коли використовується лічильники зображень, розробнику доводиться вставляти дуже вражаючий код лічильника в java-скрипт сторінки сайту.

За допомогою лічильників зображень можна збирати інформацію про більшість користувачів, але виключенням будуть пошукові роботи та користувачі з відключеними функціями використання зображень.

3.1.3 Лічильник у вигляді вбудованого модулю у сайт

Більшість сайтів зараз є динамічними – їх сторінки генеруються за принципом запиту відвідувача сайту. Динамічні сторінки сайту описуються за допомогою мов програмування, в основному PHP, ASP, JSP. Тобто, сторінка сайту сама по собі є програмою і має свої власні змінні оточення. Можна написати код мовою сайту, такою як PHP, який збирає інформацію про відвідувачів і поміщає їх у базу даних, але інформація, яка видно в результаті роботи, виводиться на виході, який зазвичай форматується як окремий файл і

вставляється на сторінки сайту, тому що такий код зручно вставляти на сторінку сайту.

Наприклад, код для програми `insert counter` (PHP-Include) у CNStats виглядає як показано на лістингу 3.2:

Лістинг 3.2 – Використання CNStats

```
... .. php - code ..
include " /usr/www/users/www.myserver.com/cnstats/cnt.php";
... .. php - code ..
```

Оскільки код лічильника включений в код сторінки, гарантується, що всі відвідувачі сайту будуть підраховані непомітно.

Однак, навіть якщо використовуються програмні вставки, є певні недоліки.:

– Інформацію про відвідувачів можна отримати лише зі змінних середовища;

– Складність визначення унікальності відвідувачів.

3.1.4 Комбінований лічильник

Код програмного модуля вставляється на сторінку динамічного сайту, і при виконанні виводиться код лічильника відвідувань зображень. У момент генерації сторінки викликається програмний код вставки і інформація, отримана зі змінної оточення, записується в базу даних.

В результаті установки програмного забезпечення на згенерованій сторінці відображається код java-скрипта для лічильника зображень. При перегляді сторінки в браузері запускається java-скрипт і викликається лічильник - зображення, з якого збирається додаткова інформація. Додаткова інформація, зібрана java-скриптом про одного і того ж користувача, додається в базу даних.

Система збору статистики дещо складніша (комбінований тип лічильників не завжди доречний), але вона надає найбільш повну інформацію про всіх відвідувачів сайту.

На даний момент представлена інформація про основні типи лічильників сесій. Виходячи з цього, ви приймаєте рішення про те, як будувати лічильник відвідувань і збирати статистику сайту.

При цьому ми враховуємо такі моменти:

– Варто використовувати універсальні програмні продукти, які підтримують різні типи лічильників-це дасть вам можливість вибирати;

– Якщо є можливість, варто використовувати комбінований лічильник, бо це останнє досягнення в плані збору статистики.

3.2 Створення програмного застосунку

Статистика про відвідувачів сайту може бути дуже корисною. Згідно зі статистикою, можна налаштувати дизайн сайту відповідно до можливостей монітора більшості користувачів, адаптувати дизайн до браузеру, використовуваному більшістю відвідувачів, скорегувати інформацію про людину, яка переглядає сайт, яку пошукову систему обрав користувач. Розглянемо, як налаштувати дизайн сайту відповідно до дозволу монітора більшості відвідувачів. Хоча деякі системи відстеження відвідувачів можуть бути дуже складними, але можна отримати цікаву інформацію про користувачів, використовуючи досить просту систему. Наведемо приклад того, як створити простий журнал відвідувань сайту за допомогою PHP і cookies.

Щоб система працювала, скрипт статистики повинен бути вбудований на кожну сторінку або на сторінку, де повинна відображатися статистика відвідувань. Скрипт записує наступні дані:

Браузер + ОС (HTTP_USER_AGENT)

IP-адреса (REMOTE_ADDR) Хост (REMOTE_HOST)

Сторінку-реферер (HTTP_REFERER) Час візиту (date("d.m.Y H : i: s")) Запрошувана адреса (REQUEST_URI)

Нижче наведено текст всього скрипта (див. лістинг 2.3):

Лістинг 3.3 – Каркас скрипта-сніфера

```
<?php
//sniffer.php
//захист від безпосереднього запуску
//скрипта стороннім користувачем
if (eregi("sniffer.php",$PHP_SELF)){
        Header("Location: index.php");
        die();
}
extract($HTTP_GET_VARS);
extract($HTTP_POST_VARS);
extract($HTTP_COOKIE_VARS);
extract($HTTP_SERVER_VARS);
```

Далі оголошуємо змінні (лістинг 3.4):

Лістинг 3.4 – Оголошення змінних для скрипта

```
$fileName="stat.txt"; //ім'я файлу із статистикою
$maxVisitors=30; //кількість записів, що відображаються
```

При перегляді статистики (лістинг 3.5):

Лістинг 3.5 – Використання куків у скрипті

```
$cookieName="visitorOfMySite"; //ім'я кукі
      $cookieValue="1"; //значення кукі
$timeLimit=86400;
```

Обмеження за часом задано в секундах, який повинен пройти з моменту останнього відвідування сайту, щоб інформація про відвідувача записалася повторно. Це значення дорівнює 1 дню, тобто один і той же користувач реєструється в статистиці 1 раз за 1 день. Якщо ця змінна дорівнює нулю, то враховуються всі відвідування одного і того ж користувача. Нижче наведені змінні для відображення статистики (див. лістинг 3.6):

Лістинг 3.6 – Змінні, що відповідають за відображення статистики

```
$header_Color="#818181";
$header_FontColor="#EFFEFF";
$font_Face="Arial, Times New Roman ";
$font_Size="2";
$tableColor="#000011";
$row_Color="#CEDECEC";
$font_Color="#0AA0A0";
$text_FontColor="#00AA0A";
```

Функція запису даних про відвідувача (див. лістинг 2.7).

Лістинг 2.7 – Функція запису даних про відвідувача

```
function saveUserData() {
GLOBAL            $fileName,      $HTTP_USER_AGENT,
                  $REMOTE_ADDR,
$REMOTE_HOST,
$HTTP_REFERER, $REQUES_URI;
$curTime=date("d.m.Y @ H : i: s"); //поточний час і дата
//готуємо дані для запису
if (empty($HTTP_USER_AGENT)){$HTTP_USER_AGENT = "Unkwnown";} if
(empty($REMOTE_ADDR)){$REMOTE_ADDR = "Not Resolved";}
if (empty($REMOTE_HOST)){$REMOTE_HOST = "Unknown";}
if (empty($HTTP_REFERER)){$HTTP_REFERER = "No Referer";} if
(empty($REQUEST_URI)){$REQUEST_URI = "Unknown";}
$data_           =   $HTTP_USER_AGENT".    ::    ".$REMOTE_ADDR".
                 :: ".$REMOTE_HOST". :: ".$     ".$HTTP_REFERER". ::
".$REQUEST_URI". ::
".$curTime".\r\n";
```

Роздільником будемо використовувати два символи ":". Наступний крок – запис зібраних даних у файл (лістинг 3.8).

Лістинг 3.8 – Запис зібраних даних у файл

```
if (is_writeable($fileName)) :
 $fp = fopen($fileName, "a");
  fputs ($fp, $data_);  fclose ($fp);
endif;
}
```

Функція запису готова. Тепер створимо функцію для виведення даних з файлу статистики (лістинг 3.9).

Лістинг 3.9 – Функція виводу даних про відвідувачів сайту

```
function showStat () { GLOBAL
$header_Color,
$headerFont_Color,
$font_Face,
$font_Size,
$table_Color,
$file_Name,
$max_Visitors,
$row_Color,
$font_Color,
$text_Font_Color;
```

Виводимо таблицю (лістинг 3.10):

Лістинг 3.10 – Формування таблиці з даними

```
$fbase=file($fileName);
$fbase = array_reverse($fbase);
$count = sizeOf($fbase);
echo "<font face=\"$fontFace\" color=\"$textFontColor\"
size=\"$fontSize\">";
echo "Усього відвідувань: $count<br><br>"; echo "<div
align=\"center\">
```

```
<table cellpadding=\"2\" cellspacing=\"1\" width=\"95%\"
border=\"0\" bgcolor=\"$tableColor\">";
echo "<tr bgcolor=\"$headerColor\"><td><font face=\"$fontFace\"
color=\"$headerFontColor\"  size=\"$fontSize\"> Браузер
</font>
</td><td>
<font face=\"$fontFace\" color=\"$headerFontColor\"
size=\"$fontSize\">IP</font>
</td> <td>
<font face=\"$fontFace\" color=\"$headerFontColor\"
size=\"$fontSize\">Хост</font></td>
<td><font face=\"$fontFace\" color=\"$headerFontColor\"
size=\"$fontSize\">Посилання</font></td>
<td><font face=\"$fontFace\" color=\"$headerFontColor\"
size=\"$fontSize\">Сторінка</font></td>
<td><font face=\"$fontFace\" color=\"$headerFontColor\"
size=\"$fontSize\">Час візиту</font></td></tr>";
echo "</font><font face=\"$fontFace\" size=\"$fontSize\">";
```

Відкриваємо файл і запускаємо цикл (лістинг 3.11):

Лістинг 3.11 – Відкриття файлу з даними про відвідувачів

```
$fbase=file($fileName);
$fbase=array_reverse($fbase);
for ($i=0; $i<$maxVisitors; $i++) :
if ($i>= sizeof($fbase)){break;}
$s = $fbase[$i];
//розділяємо
$strr = explode("::"s);
if (empty($strr)){break;}
```

Виводимо дані (лістинг 3.12):

Лістинг 3.12 – Формування сторінки звіту

```
echo "<tr><td bgcolor=\"$rowColor\"><
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[0]</font>
</td><td bgcolor=\"$rowColor\"><
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[1]</font>
</td><td bgcolor=\"$rowColor\"><
```

```
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[2]</font>
</td><td bgcolor=\"$rowColor\"><
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[3]</font>
</td><td bgcolor=\"$rowColor\"><
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[4]</font>
</td><td bgcolor=\"$rowColor\"><
font face=\"$fontFace\" color=\"$fontColor\"
size=\"$fontSize\">$strr[5]</font></td>
</tr>";
 endfor;
}
?>
```

Скрипт збору і показу статистики готовий. Тепер його треба вставити в ті сторінки, інформацію про відвідувачів якої потрібно проглянути (лістинг 3.13):

Лістинг 3.13 – Використання скрипта збору статистики у HTML-сторінці

```
<?php
include("sniffer.php");
if (! isset($$cookieName)) :
//встановити куки
setcookie($cookieName, $cookieValue, time()+$timeLimit);
saveUserData();
endif;
?>
```

Слід звернути увагу, що цей код треба вставляти в самий верх сторінки, до того, як дані передаватимуться в браузер. Інакше встановити куки не вийде. Далі зробимо сторінку, котра виводить статистику (лістинг 3.14):

Лістинг 3.14 – Формування сторінки статистики

```
<html><body>
<?php include("sniffer.php"); ?>
Статистика<br>
<?php
showStat();
?></body></html></i>
```

Підключили файл sniffer.php і викликали з нього функцію showStat (). За допомогою такого невеликого скрипта, довжиною близько 100 рядків, можна використовувати PHP для отримання і відображення статистики відвідувань сайту зручним способом. Також є можливість витягти браузер і операційну систему з HTTP_USER_AGENT і записати їх більш зручним способом.

3.3 Аналіз технічного завдання

Згідно технічного завдання розроблюване програмне забезпечення призначене для ведення обліку відвідувань сайту. Інформація, яка збирається розробленим програмним забезпеченням, повинна містити:

- ведення обліку відвідувань сайтів;
- ведення обліку відвідувань сторінок сайту;
- формування списків по IP-адресах відвідувань;
- формування списку по реферерах (сторінках, з котрих виконано перехід на сайт);
- формування списків пошукових роботів, котрі індексували сайт.

Для цього можна піти двома шляхами.

1) Створення "вбудованого" в сторінки сайту скрипта, який викликатиметься кожен раз при відкритті сторінки.

2) Аналіз лог-файлів веб-сервера.

Кожен з цих методів має свої недоліки і переваги.

Для аналізу лог-файлів веб-сервера потрібно хоча б мати доступ до цих файлів. А в реальних умовах хостингу не всюди така можливість є. Крім того, абсолютно природно, що веб-сервер Apache та IIS ведуть логи абсолютно різних форматів. А це означає, що створити "універсальний" скрипт для статистики досить проблемно, особливо, коли врахувати можливе використання і інших серверів.

Тому для вирішення поставленої задачі оберемо принцип використання

вбудованого PHP-скрипта, який виконується автоматично при завантаженні сторінки.

На даному етапі можна запропонувати концептуальну модель архітектури розробленої системи для ведення статистики відвідувань сайту.

Модель функціонування системи ведення статистики відвідувань сайту зображена на рисунку 3.1.

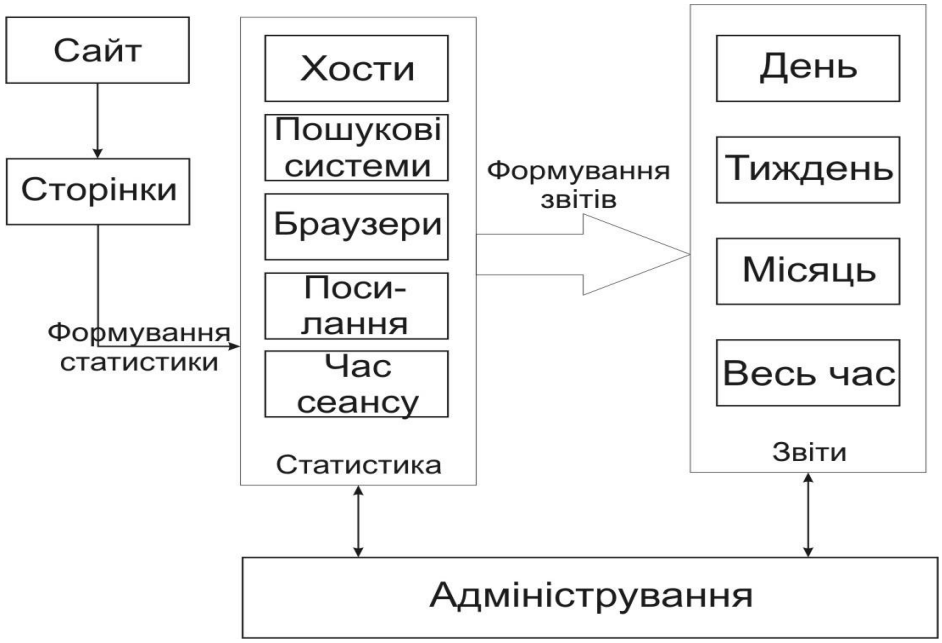

Рисунок 3.1 – Модель функціонування системи ведення статистики відвідувань сайту

З рисунка 3.1 видно, що система складається з блоків ведення статистики, формування звітів та адміністрування системи. Джерелом даних для збору є сам сайт, представлений окремими сторінками, При чому не важливо, чи це статичні сторінки, чи згенеровані динамічно в ході виконання певного сценарію чи cgi-додатку.

3.4 Створення бази даних

У створеній системі використовується СКБД mySQL, під керуванням

котрої створено базу даних для запису зібраної інформації про відвідування сайту.

Виділимо основні таблиці створеної БД:

− system_cities – таблиця для зберігання відомостей про міста та відповідні їм IP-адреси;

− system_ip – відомості про IP-адреси, з котрих заходили на сайт;

− system_ip_compact – скорочена таблиця з відомостями про IP- адреси, з котрих заходили на сайт (потрібна для формування деяких звітів);

− system_links – посилання на сайт;

− system_pages – статистика відвідування сторінок сайту;

− system_refferer – реферери на сайт;

− system_regions – таблиця для зберігання відомостей про регіони та відповідні їм IP-адреси;

− system_searchquerys – запити пошукових роботів, через котрі виходили на сайт;

− system_thits – статистика хітів (активації гіперпосилань) сайту.

Можна також сказати, що зібрана статистика архівується у допоміжних таблицях, перелік яких тут наводити не будемо.

Варто також зазначити, що хоча розроблена база даних складається з 40- ка таблиць, але реляційних зв’язків між ними немає, тобто ніякі правила цілісності посилань для розробленої БД не описано через відсутність потреби у таких реляційних зв’язках.

Відомості про назви всіх таблиць БД можна побачити в додатках.

Розглянемо структуру таблиць:

system_cities (city_id, region_id, city_name);

system_ip (id_ip, ip, putdate, id_page, browsers, systems);

system_ip_compact(init_ip, end_ip, city_id); system_links (id_links, name, comment);

system_pages (id_page, name, title, id_site);

system_refferer (id_refferer, name, putdate, ip, ip_page);

system_regions (region_id, region_name);

system_searchquerys (quer_id, query, putdate, ip, ip_page, searches);

system_thits (hits).

Для створення бази даних запустимо на виконання SQL-скрипт для СКБД mySQL, текст котрого наведено у Додатку А.

3.5 Розробка інформаційної системи

Приступимо до реалізації системи. Для створення додатку використали VS Code. Адже даний редактор дозволяє використовувати різні мови програмування, а також у режимі реального часу переглядати створенні сторінки проекту. За допомогою VS Code можна створювати не тільки статичні сторінки, але також редагувати файли типу PHP, використовувати різноманітні клієнтські та серверні скрипти.

Перш ніж приступити до розробки користувальницького інтерфейсу, розглянемо функціональну схему системи. Схема повинна бути розрахована на всі елементи інтерфейсу, які використовувались під час розробки системи. Функціональна схема показана на рис. 3.1. На концептуальному рівні представлення роботи схема відображає загальні принципи застосування даної системи.

В розробленій системі можливість збору інформації та її запису в базу даних реалізована за допомогою скрипта count.php. Він вбудований у сторінки, за якими треба слідкувати, тобто відстежувати кількість переглядів . Принцип роботи скрипта та методи його використання наведено у розділі 3.6.

Оскільки деякі сценарії управління системою відповідають за створення звітів, структурну схему системи можна переглянути на рис. 3.2.

Деякі системні модулі не представлено на схемі, але їх текст можна побачити в Додатку Б.

3.6 Принцип роботу модулю для збору статистики

У проекті представлено модуль для збору статистики трафіку сайту. Це вбудований в сайт лічильник, але він записує не тільки кількість користувачів, які перейшли за певним гіперпосиланням, але і багато іншої корисної інформації про відвідувачів сайту. Написаний мовою програмування PHP, цей лічильник відвідувань є досить потужною системою для збору та аналізу інформації для подальшої аналітики.

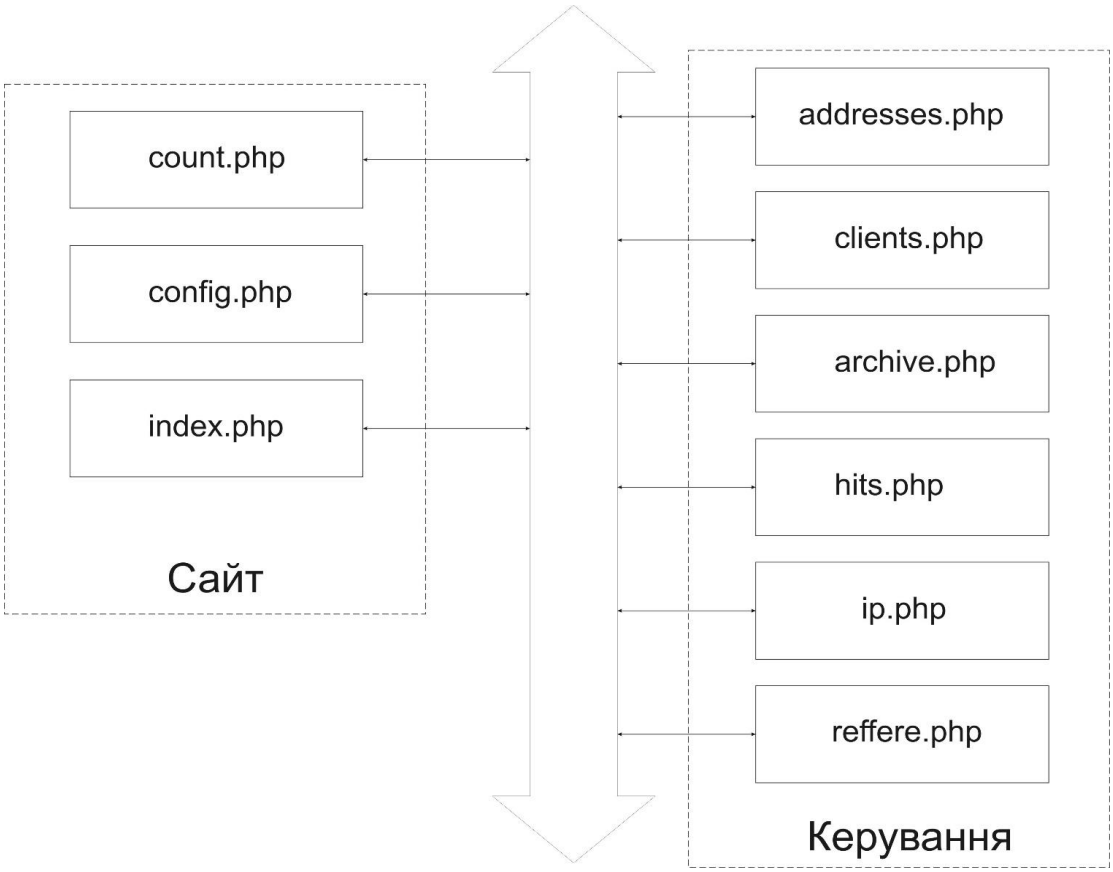

Рисунок 3.2 – Структурна схема системи

Це не просто лічильник відвідувань, а інструмент, який дозволяє відстежувати потоки відвідувачів на сайті, скільки та які сторінки вони переглядають, скільки часу перебувають на сайті, звідки вони заходять на сайт, з інших сайтів або з пошукових систем. Також є можливість представлення даних у вигляді кругових діаграм та графіків.

Перевага вбудованого лічильника полягає в тому, що він фіксує всіх відвідувачів, а не лише людей, які завантажили зображення банера, на відміну від зовнішнього лічильника.

Тому важливо знати, чи відвідувався сайт пошуковими системами або Користувач записав весь сайт цілком на свій комп'ютер за допомогою менеджера завантажень.

Лічильник відвідувань збирає статистику відвідувань сайту і відображає інформацію про загальну і підраховану кількість хостингів, загальну і підраховану кількість переглядів, а також про операційну систему і браузер, що використовуються відвідувачами, як для окремих сторінок, так і для всього сайту в цілому. Реєструються відвідування роботами найвідоміших пошукових систем і проіндексованих сторінок.

Лічильник відвідувань надає можливість додавати адреси ресурсів, що містять посилання на сайти, і відстежувати кількість відвідувань з них. Інформація надається в 8 часових інтервалах: "сьогодні", "вчора", "7 днів тому", "30 днів тому", "60 днів тому", "90 днів тому", "у минулому році" і " за весь час". Більшість звітів містять щоденну, щотижневу та щомісячну статистику.

Навіть на таких відвідуваних сайтах розмір бази даних не перевищує 2 мб. Це пов'язано з конфігурацією бази даних. Повна інформація зберігається всього 1 день, потім стискається і поміщається в таблицю щоденного архіву. Через тиждень інформація стискається в таблицю за тиждень, а через місяць - в таблицю за місяць. Зрозуміло, що вся інформація, яка не потрібна для виведення

звіту, буде видалена. Звичайно, це досягається за рахунок деякого скорочення функціональності, але цього можна уникнути. Крім того, таблиці з іменами змінних можуть використовуватися в розвинених статистичних системах.

Ведеться облік браузерів Chrome, Firefox, Safari, Edge та інші, розширена таблиця pages, для поступового переходу з URL на назви сторінок. Тепер додавши на сторінку назву в змінній $titlepage замість URL, будуть виводиться назви сторінок.

Статистика за IP-адресами показує щоденні, щотижневі та щомісячні звіти з сторінковою навігацією. Є щоденна, щотижнева та щомісячна статистика рефералів, а також статистика глибини та часу відвідування.

Також ведеться статистика по країнах і містах, з яких користувачі відвідували сайт.

3.7 Встановлення модулю для збору статистики

Для того, щоб модуль відпрацьовував без помилок, створили базу даних (за замовчуванням count), в якій розмістили таблиці з файлу PowerCoutner.sql. Залежно від хостингу, користувач має різні інструменти для створення бази даних. Це можна зробити за допомогою веб-інтерфейсу або шляхом виконання SQL-запитів для створення бази даних (create database count;).

Підключення до бази даних налаштовується у файлі адміністратора (admin/config.php), який містить адресу сервера MySQL ($dblocation) і повинен містити ім'я бази даних (dbdbname), ідентифікатор користувача (user dbuser) та його пароль ($dbpasswd). Усі 4 змінні повинні бути надані хостингом. Для того, щоб сторінки на сайті могли отримувати доступ до лічильника і вести статистику, вам необхідно додати інструкції в файл на початку сторінки. count.php (лістинг 3.15):

Лістинг 3.15 – Використання лічильника

```
<?php
      include "count.php";
?>
```

Як показано в індексі тестової сторінки index.php. Користувач не обмежений у виборі розташування цієї структури. Зовсім не важливо, чи завантажить відвідувач сторінку до кінця – все одно буде враховуватись. Це відбувається тому, що PHP-код виконується на сервері, і клієнту нічого не надсилається, поки він не буде виконаний. Отже, якщо відвідувач отримує лише перший байт, він уже зараховується.

При першому зверненні відвідувача до сторінки в таблиці (pages) сторінок створюється запис, відповідна цій сторінці, яка зберігається в розроблюваної системі статистики для відображення. Кількість сторінок не обмежена.

Якщо помістити назву сторінки у змінну $titlepage у системному звіті перед включенням файлу за допомогою інструкції require_once, ця сторінка буде об'єднана з цією назвою. Крім того, ви можете згрупувати кілька сторінок в один рядок, присвоївши їм однакове ім'я (лістинг 3.16).

Лістинг 3.16 – Робота з декількома сторінками

```
<?php
     $     $titlepage = "Назва
      сторінки";
      require_once("count.php");
?>
```

Окремо слід зазначити, що робоча таблиця архівується у вигляді щоденних, щотижневих і щомісячних таблиць. Стиснення відбувається о 00: 00 при першому доступі до сторінки адміністрування. Однак є можливість примусово стиснути дані в системі. Наприклад, за цей процес відповідає по cron – скрипт admin/archive.php.

Можна архівувати дані самостійно, перейшовши за посиланням на Cron. Однак, якщо це взагалі не потрібно, достатньо час від часу входити в систему управління, і система буде працювати сама по собі.

3.7.1 Помилки з якими можна зіткнутись

Слід пам'ятати, що сервер не є клієнтською машиною Windows XP, і вони та їх налаштування часто відрізняються один від одного. Тому дуже ймовірно, що виникне проблема, в основному пов'язана з нестандартною конфігурацією змінної сервера. Так що якщо щось не враховано і не береться до уваги, потрібно проаналізувати це за допомогою PHP-функції php_info() і порівняти ідентифікатор змінної оточення з таким же ідентифікатором кількості файлів. Якщо виявлено невідповідність, виправите її за допомогою змінної, зазначеної у звіті php_info().

За замовчуванням лічильник підраховує сторінки з різними параметрами лише для кожної сторінки:

```
index.php?id=1
index.php?id=2
```

Вони вважаються однією і тією ж сторінкою (index).php), за якою слідують всі результати. Щоб такі сторінки вважалися різними, необхідно замінити весь лічильник в коді: $_SERVER['PHP_SELF'] на $_SERVER['REQUEST_URI'] – достатньо відкрити кожен файл в блокноті та скористатися функцією автозаміни.

4 ІНСТРУКЦІЯ КОРИСТУВАЧА

Відстежувати популярність сторінок за допомогою лічильника на сторінці адміністрування розташованою в каталозі admin (файл index.php). Головна сторінка (на рис. 4.1) містить список всіх сторінок, які включені в збір статистики, а на кожній сторінці відображається різна статистика відвідувань сайту.

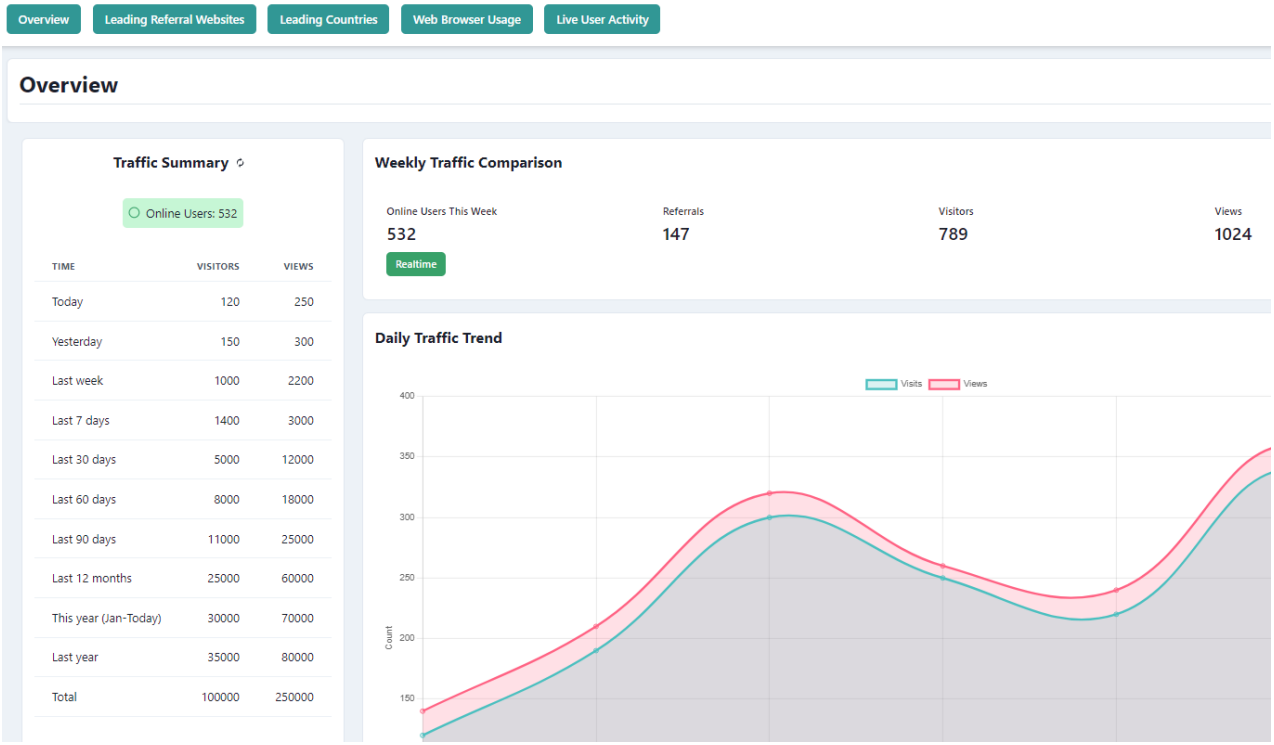

Рисунок 4.1 – Головна сторінка

Вгорі знаходиться меню з посиланням на сторінки. Коли ви натискаєте на меню, ви можете переглянути статистику по всьому сайту, але коли ви натискаєте на посилання з таблиці, ви можете побачити статистику по кожній конкретній сторінці.

Якщо перейти на головну сторінку, можна побачите таблицю, яка

показує кількість відвідувань та хостингів за 5 часових інтервалів: "сьогодні", "вчора", "7 днів", "30 днів", "60 днів", "90 днів", "12 місяців", "у минулому році" та "усього" на цій сторінці сайту (рис. 4.2).

Traffic Summary

Online Users: 532

TIME	VISITORS	VIEWS
Today	120	250
Yesterday	150	300
Last week	1000	2200
Last 7 days	1400	3000
Last 30 days	5000	12000
Last 60 days	8000	18000
Last 90 days	11000	25000
Last 12 months	25000	60000
This year (Jan-Today)	30000	70000
Last year	35000	80000
Total	100000	250000

Рисунок 4.2 – Таблиця трафіку

Також на банері Головної сторінки розташована копка "Weekly Traffic Comparison" (див. рис. 4.3), перехід по якій призводить до сторінки розподілу хостів і хітів по яким зайшли користувачі на сайт.

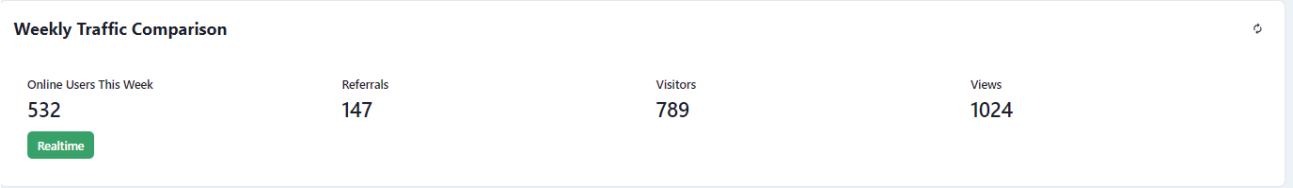

Рисунок 4.3 – Банер з трафіком

Також на головній сторінці присутній графік "Daily Traffic Trend". На ньому зображено кількість відвідувань та переглядів на сайті в реальному часі (див. рис. 4.4).

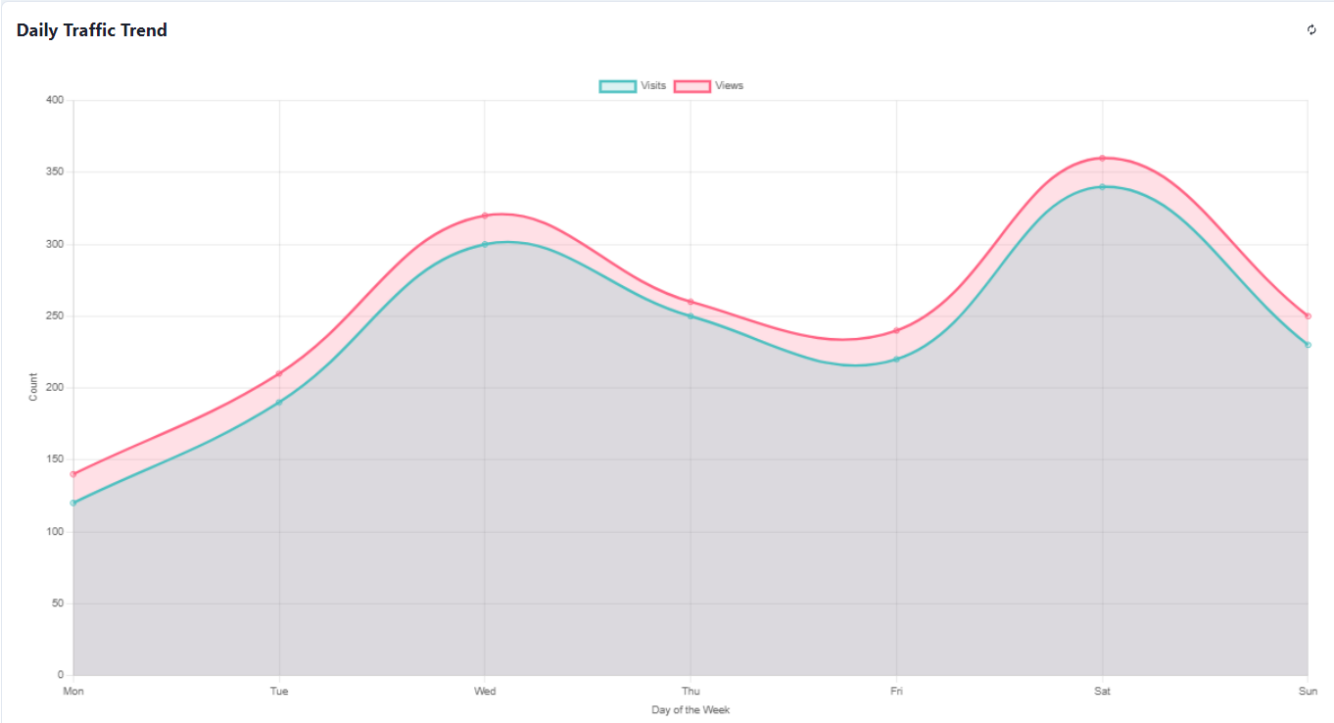

Рисунок 4.4 – Графік відвідувань

Меню переходу по сторінкам складається з наступних пунктів:

– " Overview " (головна сторінка на якій розташована таблиця загального графіку по часовим проміжкам, кількість онлайн користувачів та графік відвідувачів і переглядів);

– "Leading Referral Websites" (сторінка з таблицею на якій розміщується рейтинг популярних рефералів та кількість переходів);

– "Leading Countries Visiting" (сторінка з таблицею рейтингу країн та кількістю відвідувачів з цих країн);

– "Live User Activity" (сторінка з таблицею в якій знаходиться інформація про користувачів);

– "Web Browser Usage Statistics" (сторінка с таблицею та діаграмою популярних веб-браузерів, які використовувались відвідувачами);

Перехід за посиланням "Leading Referral Websites" відкриває сторінку, де розташована таблиця популярності рефералами, в якій міститься інформація про сайти\соціальні мережі з яких користувачі потрапили на сайт. Загальний вигляд сторінки зображено на рис. 4.5.

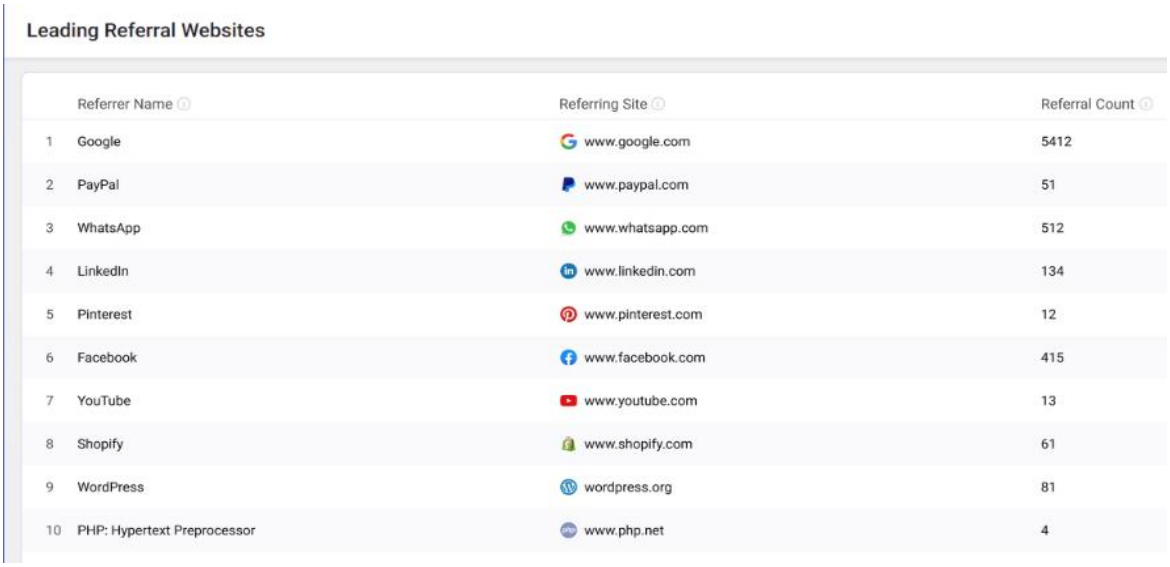

Рисунок 4.5 – Сторінка з рефералами

Перехід за посиланням "Leading Countries" приводить до сторінки з таблицею, в якій вказано список країн та кількість зарахованих користувачів, які відвідали сайт. Загальний вигляд сторінки зображено на рис. 4.6.

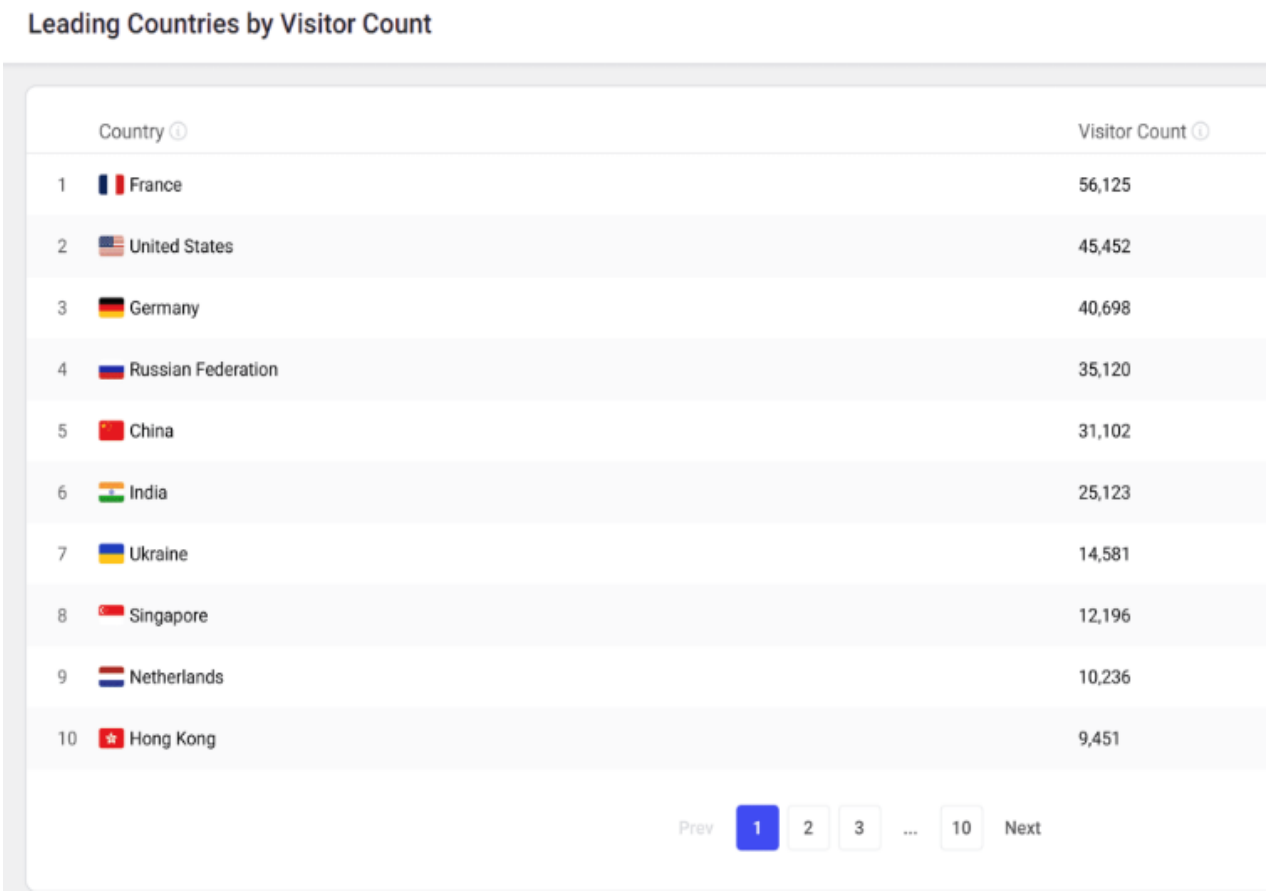

Рисунок 4.6 – Сторінка з рейтингом країн

Перехід до сторінки " Live User Activity" призводить до сторінки з таблицею, в якій вказана інформація про відвідувачів. Така як: браузер, яким користуються, країна та місто проживання, сторінка яку відвідали та час, який було витрачено на її перегляд, айді користувача, пошта та роль, якщо користувач зареєстрован в системі. Загальний вигляд сторінки зображено на рис. 4.7.

Live User Activity

BROWSER	COUNTRY	CITY	ONLINE FOR	PAGE	REFERRER	USER
Chrome	USA	New York	00:05:00	/dashboard	google.com	101, user1@example.com, Admin
Firefox	Canada	Toronto	00:08:00	/home	example.com	102, user2@example.com, Editor
Safari	Australia	Sydney	00:03:00	/settings	yahoo.com	103, user3@example.com, Viewer
Edge	United Kingdom	London	00:15:00	/profile	bing.com	104, user4@example.com, Moderator
Opera	Ukraine	Lvive	00:10:00	/products	yandex.ru	105, user5@example.com, Admin
Chrome	Brazil	São Paulo	00:22:00	/checkout	facebook.com	106, user6@example.com, Editor
Firefox	Germany	Berlin	00:07:00	/about	linkedin.com	107, user7@example.com, Viewer
Safari	India	Mumbai	00:30:00	/contact	twitter.com	108, user8@example.com, Moderator
Internet Explorer	China	Shanghai	00:05:00	/faq	weibo.com	109, user9@example.com, Admin
Edge	South Africa	Cape Town	00:12:00	/services	google.co.za	110, user10@example.com, Editor

Рисунок 4.7 – Сторінка з інформацією про користувачів

На сторінці " Web Browser Usage Statistics" розташована таблиця та кругова діаграма (рис. 4.8). Зміст таблиці передає інформацію про веб-браузери які використовувались найчастіше за все впродовж 30 днів. Після цього періоду дані оновлюються автоматично. Приклад таблиці наведено на рис. 4.9.

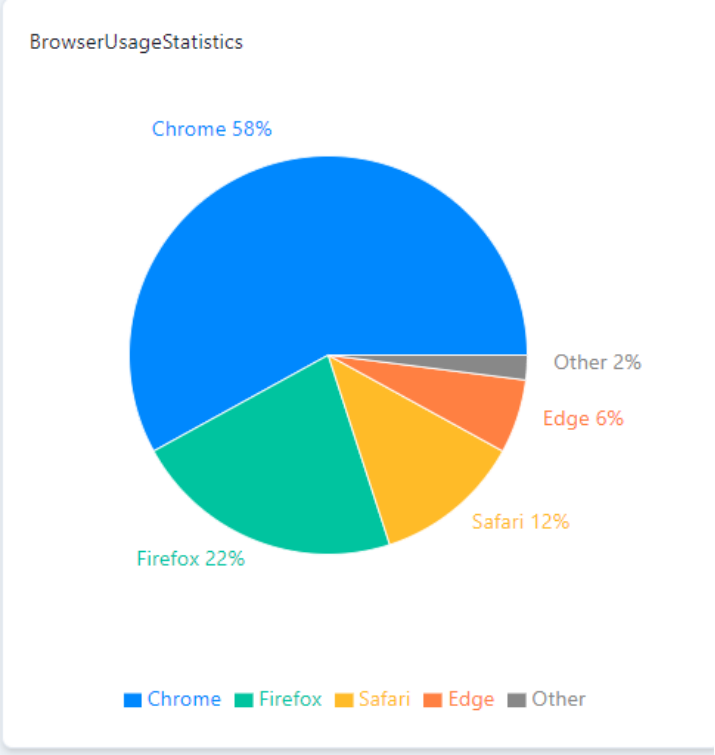

Рисунок 4.8 – Кругова діаграма

Statistics for Browsers in the Past 30 Days

VISITOR'S BROWSER	VISITORS	PERCENT SHARE
Chrome	580	58%
Firefox	220	22%
Safari	120	12%
Edge	60	6%
Other	20	2%

Рисунок 4.9 – Таблиця з веб-браузерами

Крім того можна скачувати звіті, вони будуть розташовані у текстовому форматі (див. рис. 4.10).

Рисунок 4.10 – Імпортовані дані

Але отримані дані легко імпортувати та відкрити за допомогою Excel, що значно спрощує проведення аналітичних тестів, побудову графіків, таблиць і виконання різноманітних операцій з вибіркою даних. Використання Excel дозволяє швидко отримувати необхідну інформацію, виявляти тренди та патерни, а також здійснювати порівняльний аналіз даних. Це робить процес обробки та аналізу даних більш зручним та ефективним.

5 ПРОВЕДЕННЯ АНАЛІТИЧНИХ ТЕСТІВ

Система для ведення статистики відвідування сайту службує засобом для збору статистичних даних про людей які потрапили на сайт, чи просто їм користуються. Збирає такі данні як: країна\місто проживання, айді та пошту користувача, якщо він авторизован на нашому сайті, браузер, який використовувався, час який користувач провів на певній сторінці. Використовуючи отриманні дані за допомогою застосунку Orange проаналізуємо та побудуємо діаграми.

Orange - це візуалізації даних, які допомагають виявити закономірності даних, надають інтуїтивне розуміння процедур аналізу даних або підтримують комунікацію між аналітиками даних та експертами в предметній області. Віджети візуалізації включають діаграму розсіювання, секторну діаграму та гістограму, а також візуалізації для конкретних моделей, наприклад, дендрограму, силуетну діаграму та деревоподібну візуалізацію. Багато інших візуалізацій доступні в надбудовах і включають візуалізацію мереж, хмар слів, географічних карт тощо.

Orange включає в себе багато стандартних візуалізацій. Діаграма розсіювання чудово підходить для візуалізації кореляцій між парою атрибутів, секторна діаграма - для відображення базової статистики, теплова карта - для огляду всього набору даних, а проекційні діаграми, такі як метод MDS, - для побудови двовимірних мультиплікативних даних [8].

Робоче середовище програми Orange показано на рис. 5.1:

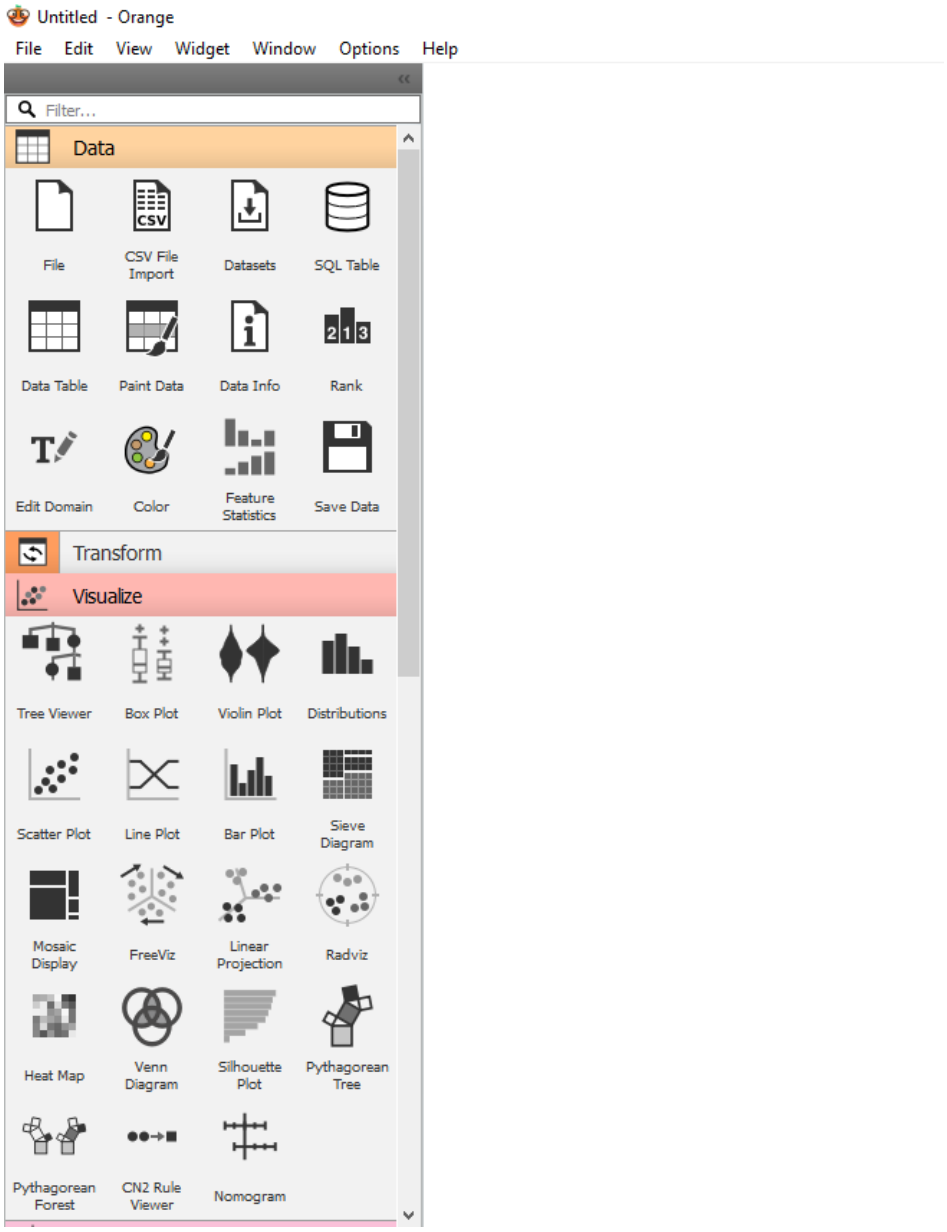

Рисунок 5.1 – Робочий інтерфейс програми

На рисунку показані інструменти, що використовуються для обробки даних, які проводили тест, перетворення даних та візуалізацію даних. Orange надає інтуїтивно зрозумілий інтерфейс для побудови робочих процесів аналізу даних з використанням модулів, які можна легко перетягувати і підключати один до одного. За допомогою цієї програми можна зручно імпортувати дані з різних джерел, застосовувати численні методи попередньої обробки та аналізу та

створювати інтерактивні візуалізації для кращого розуміння результатів. Завдяки різноманітності інструментів і модулів Orange забезпечує потужну підтримку для вирішення складних завдань в області аналізу даних і машинного навчання.

Вхідні дані мають такий вигляд (на рис. 5.2):

URL	Pageviews	Unique_Pageviews	Avg_Time_on_Page	Entrances	Bounce_Rate	%_Exit	OBJECTID
/opendata.dc.gov/	426,483	167,594	0:00:31	97,865	13.50%	10.05%	0
/opendata.dc.gov/datasets/residential-parking-p	32,604	18,550	0:01:03	1,576	20.74%	15.77%	1
/opendata.dc.gov/datasets?sort_by=name	14,931	8,116	0:00:41	769,000	4.22%	8.32%	2
/opendata.dc.gov/datasets?keyword=transporta	13,416	7,135	0:00:55	346,000	5.91%	9.90%	3
/opendata.dc.gov/datasets?keyword=property	11,130	5,829	0:00:49	166,000	5.92%	7.86%	4
/opendata.dc.gov/datasets?keyword=demograp	10,358	5,454	0:00:39	164,000	8.70%	7.50%	5
/opendata.dc.gov/datasets?keyword=education	10,253	5,395	0:00:33	198,000	11.40%	7.08%	6
/opendata.dc.gov/404	10,085	5,152	0:00:39	3,336	25.30%	21.20%	7
/opendata.dc.gov/datasets	9,218	6,590	0:00:36	721,000	14.66%	10.18%	8
/opendata.dc.gov/datasets?keyword=political	8,151	4,528	0:00:56	156,000	11.79%	10.01%	9
/opendata.dc.gov/datasets?keyword=environme	7,734	4,117	0:00:53	140,000	7.03%	8.75%	10
/opendata.dc.gov/datasets?keyword=economic	6,433	3,551	0:00:52	93,000	10.18%	7.93%	11
/opendata.dc.gov/datasets/b761e3f584344ae1b	5,973	5,259	0:09:10	5,037	11.11%	83.02%	12
/opendata.dc.gov/datasets?sort_by=relevance	5,636	2,816	0:00:04	2,718	0.88%	1.44%	13
/opendata.dc.gov/datasets?keyword=health	5,524	3,033	0:00:26	80,000	11.67%	5.81%	14
/opendata.dc.gov/datasets?keyword=imagery	5,492	2,434	0:00:17	211,000	4.05%	4.59%	15
/opendata.dc.gov/datasets?keyword=cultural	5,454	3,041	0:00:30	68,000	11.11%	5.87%	16
/opendata.dc.gov/datasets?keyword=planning	5,282	2,905	0:00:41	92,000	6.62%	6.91%	17
/opendata.dc.gov/datasets?keyword=public saf	5,006	2,217	0:00:04	50,000	0.00%	0.76%	18
/opendata.dc.gov/datasets?keyword=location	4,916	2,746	0:00:27	72,000	16.00%	5.92%	19
/opendata.dc.gov/datasets?keyword=structure	4,727	2,563	0:00:44	69,000	8.18%	6.62%	20
/opendata.dc.gov/datasets/residential-parking-p	4,065	3,074	0:01:47	30,000	26.67%	19.26%	21
/opendata.dc.gov/datasets?keyword=elevation	3,955	2,141	0:00:52	79,000	12.17%	9.05%	22
/opendata.dc.gov/datasets?keyword=Financials	3,751	1,971	0:00:27	63,000	15.22%	5.33%	23
/opendata.dc.gov/datasets/bda20763840448b5	3,673	1,800	0:00:11	1,249	2.23%	6.97%	24
/opendata.dc.gov/datasets?keyword=utility	3,652	1,986	0:00:36	43,000	11.76%	7.06%	25
/opendata.dc.gov/datasets?keyword=historic	3,514	1,970	0:00:31	54,000	6.67%	6.23%	26
/opendata.dc.gov/datasets?q=transportation	3,362	2,283	0:00:46	84,000	14.15%	9.82%	27
/opendata.dc.gov/datasets/crime-incidents-in-2	3,316	2,291	0:02:10	1,512	35.66%	47.59%	28
/opendata.dc.gov/datasets?keyword=Performar	3,057	1,768	0:00:15	55,000	13.85%	4.51%	29
/opendata.dc.gov/datasets?q=property	2,993	1,910	0:00:48	70,000	14.61%	8.12%	30
/opendata.dc.gov/datasets?keyword=safety	2,960	1,966	0:00:53	88,000	8.65%	12.23%	31
/opendata.dc.gov/datasets?keyword=recreation	2,938	1,655	0:00:40	32,000	9.84%	6.71%	32
/opendata.dc.gov/datasets?q=demographic	2,421	1,535	0:00:44	51,000	17.74%	8.26%	33

Рисунок 5.2 – Таблиця представлення даних

Вони представлені у вигляді таблиці, в якій знаходяться назва сторінки, загальна кількість переглядів, та кількість переглядів зареєстрованих користувачів, час, який користувач провів на сторінці та кількість натискань на сторінці.

Імпортували отриманні дані в програму Orange, та побудували схему відображення графіків (на рис. 5.3).

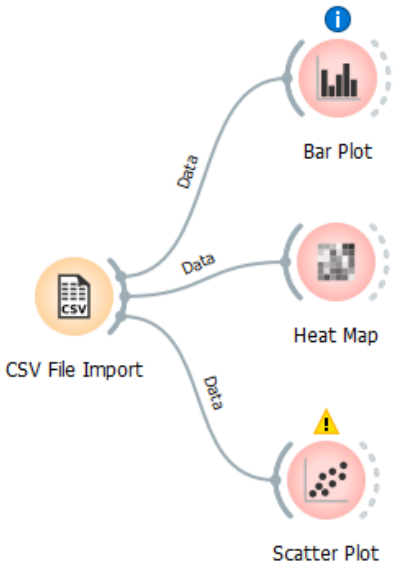

Рисунок 5.3 – Загальна схема

Створили графік розсіювання, як показано на рис.5.4

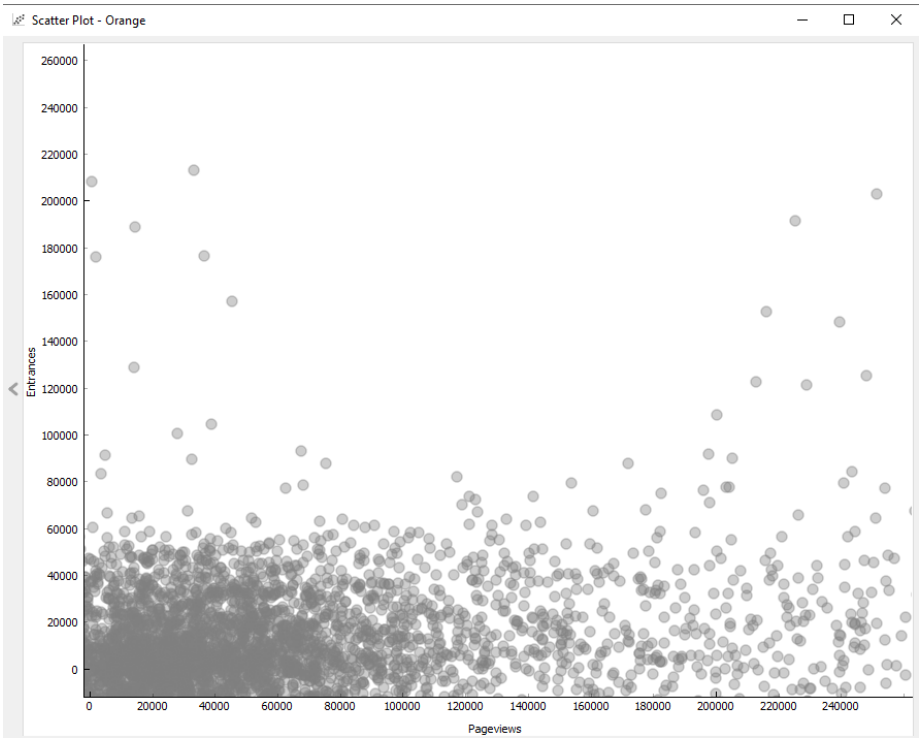

Рисунок 5.4 – Графік розсіювання

Такий графік ілюструє відношення переглядів сторінок до загальної кількості відвідувань користувачами нашого сайту. Можна побачити, що більшість переглядів сторінок приходиться на початок графіку. Тобто виходячи з цього, можна сказати, що більшість користувачів затримуються на головній сторінці сайту, а меншість доходить до віддалених сторінок, наприклад, як оформлення заказу на сайті.

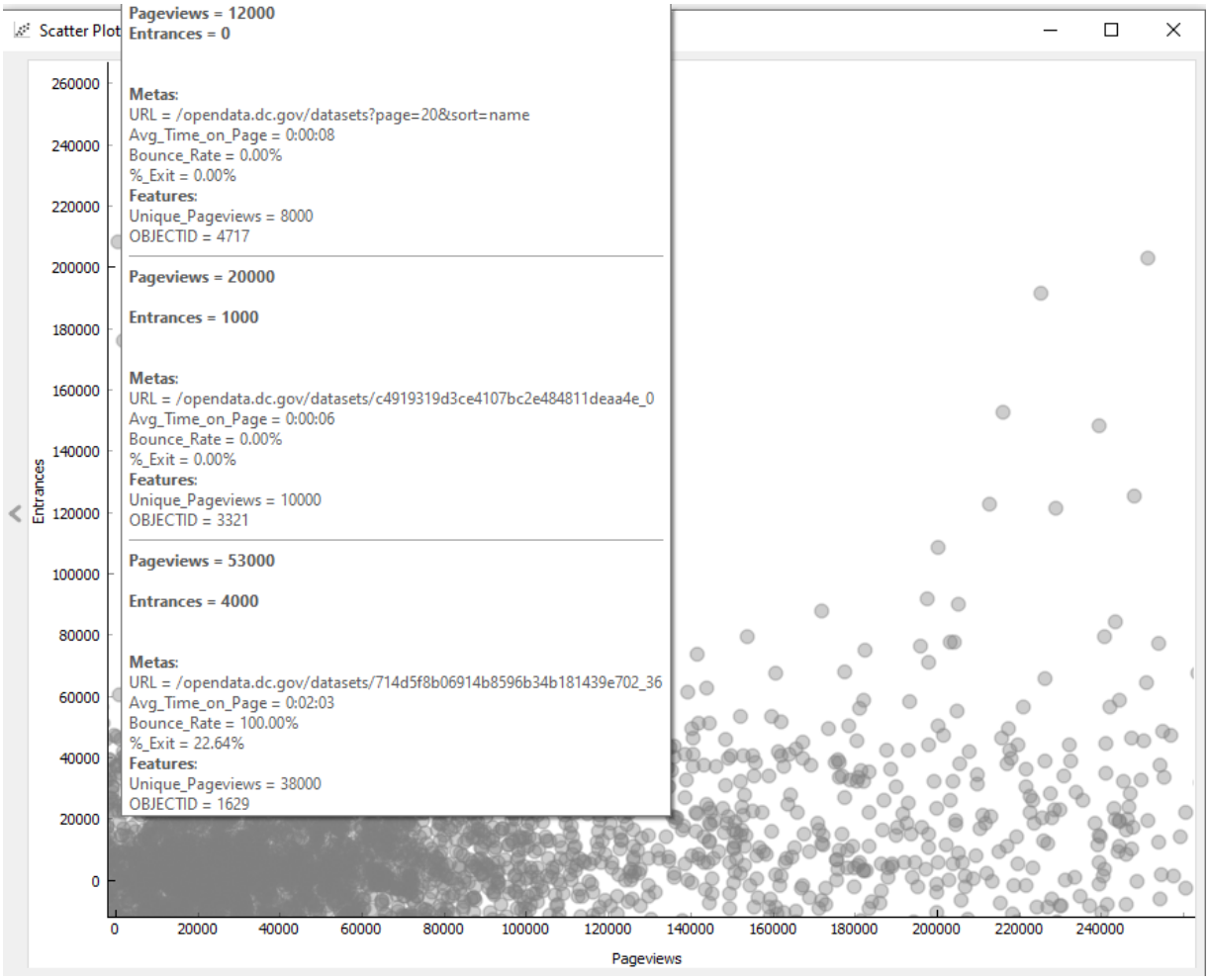

Рисунок 5.5 – Приклад відображення графіку

При наведенні мишкою на бульбашку, то можна отримати більш детальну інформацію, таку як: адресу сторінки, час проведення користувачем на сторінці та кількість користувачів які зареєстровані на сайті (див.рис.5.5).

На наступному графіку проставлено відношення кількості переглядів сторінок сайту до часу, який витрачено в середньому користувачем на перебуванні на певній сторінці. Виходячи з результатів можна сказати, то чим глибше користувач переміщується по сайту, тим менше часу він витрачає на перегляд сторінок (рис.5.6).

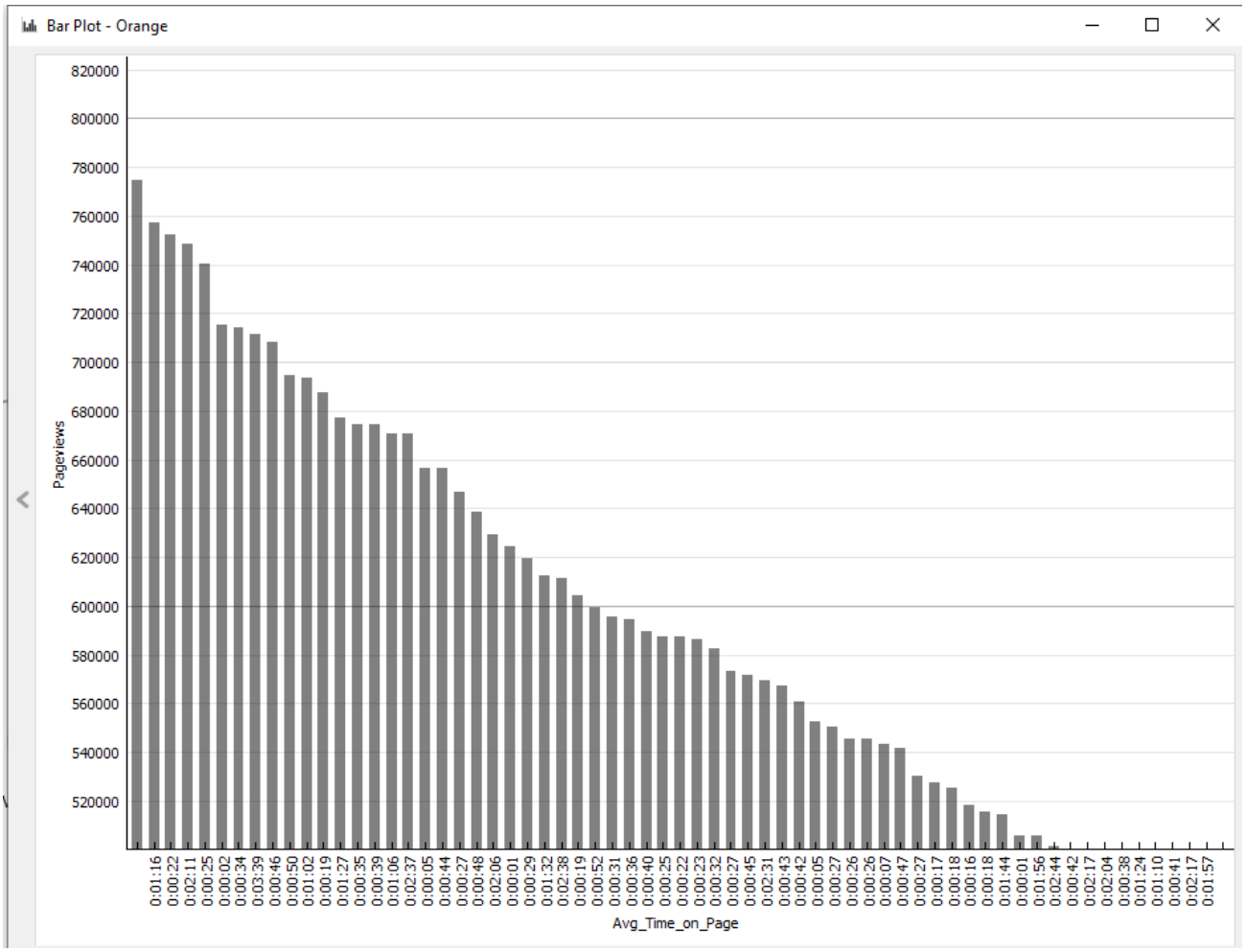

Рисунок 5.6 – Стовпчастий графік

Наприклад, такий графік може демонструвати таку ситуацію, коли на вибір товару значно більше часу знадобиться ніж на перегляд сторінки про компанію.

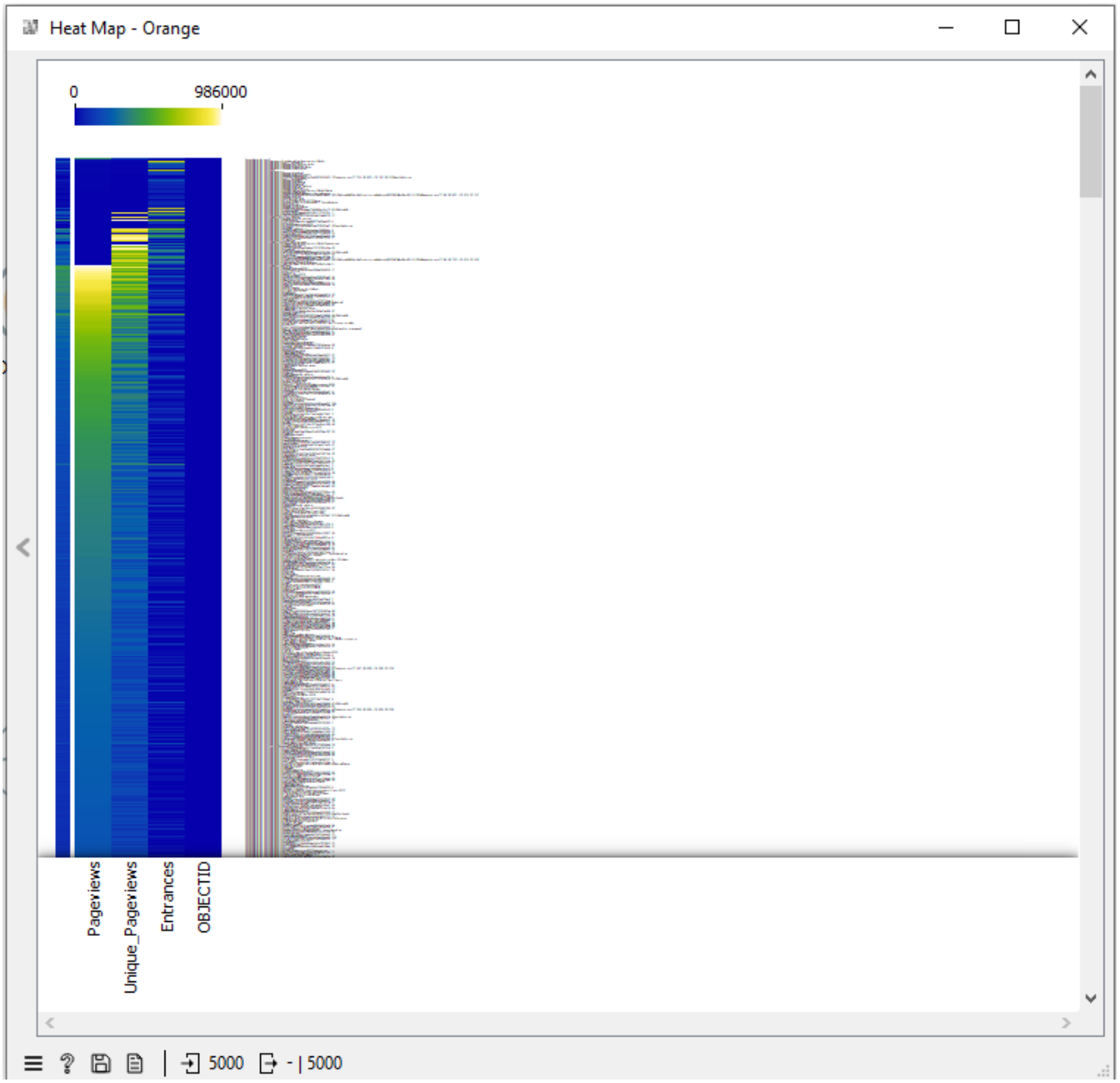

Рисунок 5.7 – Теплова діаграма

На рисунку 5.7 представлена теплова діаграма. Вона ілюструє в залежності від кількості переглядів, на які сторінки частіше за все заходять відвідувачі. Чим ближче колір на діаграмі наближається до жовтого, тим більша кількість користувачів зупинялась на певній сторінці.

ВИСНОВОК

У рамках роботи здійснено огляд можливостей програмного забезпечення для вирішення поставленого завдання. Проведено аналіз існуючих систем збору статистики відвідувань сайтів, а також розглянуто методи вирішення проблем, пов'язаних зі збором і аналізом цих даних. Це дало змогу визначити основні вимоги до створюваного модуля для збору статистичних даних.

На основі аналізу обґрунтовано вибір проектних рішень для створення інформаційно-програмного забезпечення. Розроблено технології збору, передачі, підготовки та обробки інформації. Для практичної реалізації обрали мову програмування PHP, що дозволило створити вбудовані PHP-скрипти для розрахунку статистики використання для кожної сторінки сайту. Це рішення дозволяє використовувати систему без додаткових витрат на дослідження та забезпечує простоту використання розробленого модуля.

Розроблений модуль для ведення статистики відвідувань веб-сайтів є ефективним інструментом для збору та візуалізації даних про відвідувачів. Він забезпечує власникам веб-ресурсів цінну інформацію для покращення контенту, структури та дизайну сайту, а також для оптимізації маркетингових стратегій. Завдяки своїй простоті, гнучкості та можливостям розширення, модуль може бути корисним для різних типів веб-сайтів і сприяти їхньому розвитку та успіху.

Для подальшого розвитку модуля можна додати:

− Теплові карти та запис сеансів для глибшого аналізу поведінки користувачів.

− Автоматичні звіти та сповіщення для оперативного реагування на важливі події.

− Аналіз продуктивності сторінок та A/B тестування для оптимізації контенту та інтерфейсу.

− Мобільні додатки для зручного доступу до статистики на ходу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Пошукові системи: їх роль і вплив у сучасному інтернет-просторі. Webmate.ua. – Режим доступу: https://webmate.ua/poshukovi-sistemi-istoriya-rozvitok-ta-suchasni-tendenciyi.
2. Бойко Соломія. SEO та Digital-маркетинг: ключові стратегії для успішного просування вашого бізнесу в Інтернеті. Ми Вінничани. – Режим доступу: https://vinnychany.info/ukraine/seo-ta-digital-marketynh-kliuchovi-stratehii-dlia-uspishnoho-prosuvannia-vashoho-biznesu-v-interneti/?gad_source=1.
3. https://www.facebook.com/SendPulseua. Що таке пошукова оптимізація: визначення | SendPulse UA. SendPulse. – Режим доступу: https://sendpulse.ua/support/glossary/seo-optimization.
4. Сергей Шевченко. Основні інструменти Веб-аналітики: з чого почати?! Adwservice. – Режим доступу: https://adwservice.com.ua/uk/osnovni-instrumenty-veb-analityky-z-chogo-pochaty.
5. HTML і CSS довідник українською. Html CSS довідник. – Режим доступу: https://html-css.co.ua/.
6. Node.js — Run JavaScript Everywhere. Nodejs.org. – Режим доступу: https://nodejs.org/en.
7. Що таке React JS і для чого він потрібен? - DAN.IT. Dan-it.com.ua. – Режим доступу: https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen .
8. Bioinformatics Laboratory, University of Ljubljana. Orange Data Mining. Orange Data Mining. – Режим доступу: https://orangedatamining.com .

ДОДАТОК А

SQL-запити на створення бази даних

```
CREATE TABLE system_ip (
id_ip int(32) NOT NULL auto_increment, ip bigint(11) NOT NULL
default '0',
putdate datetime NOT NULL default '0000-00-00 00:00:00', id_page
int(10) NOT NULL default '0',
browsers
enum('none','msie','opera','netscape','firefox','myie','mozilla')
NOT NULL default 'none',
systems
enum('none','windows','unix','macintosh','robot_yandex','robot_googl
e','robot_r ambler','robot_aport','robot_msnbot') NOT NULL default
'none',
PRIMARY KEY  (id_ip),
KEY putdate (putdate), KEY ip (ip)
) TYPE=MyISAM;
CREATE TABLE system_links (
id_links int(8) NOT NULL auto_increment, name text,
comment text,
PRIMARY KEY  (id_links)
) TYPE=MyISAM;
CREATE TABLE system_pages (
id_page int(10) NOT NULL auto_increment, name text,
title text,
id_site int(4) default NULL, PRIMARY KEY  (id_page)
) TYPE=MyISAM;
CREATE TABLE system_refferer (
id_refferer int(16) NOT NULL auto_increment, name text NOT NULL,
putdate datetime NOT NULL default '0000-00-00 00:00:00', ip
bigint(11) NOT NULL default '0',
id_page int(8) NOT NULL default '0', PRIMARY KEY  (id_refferer),
KEY id_page (id_page), KEY ip (ip)
) TYPE=MyISAM;
CREATE TABLE system_searchquerys (
quer_id int(11) NOT NULL auto_increment, query tinytext NOT NULL,
putdate datetime NOT NULL default '0000-00-00 00:00:00', ip
bigint(20) NOT NULL default '0',
id_page int(11) NOT NULL default '0',
searches enum('yandex','google','rambler','aport','mail','msn') NOT
NULL default 'yandex',
PRIMARY KEY  (quer_id)
) TYPE=MyISAM;
CREATE TABLE system_thits ( hits int(11)
) TYPE=MyISAM;

CREATE TABLE system_arch_hits (
```

```
id_ip int(3) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00',
hosts_total int(11) NOT NULL default '0',
host int(11) NOT NULL default '0', hits_total int(11) NOT NULL
default '0', hits int(11) NOT NULL default '0', PRIMARY KEY
(id_ip)
) TYPE=MyISAM;
CREATE TABLE system_arch_hits_month ( id_ip int(3) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00',
hosts_total int(11) NOT NULL default '0',
host int(11) NOT NULL default '0', hits_total int(11) NOT NULL
default '0', hits int(11) NOT NULL default '0', PRIMARY KEY
(id_ip)
) TYPE=MyISAM;
CREATE TABLE system_arch_hits_week ( id_ip int(11) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00',
hosts_total int(11) NOT NULL default '0',
host int(11) NOT NULL default '0', hits_total int(11) NOT NULL
default '0', hits int(11) NOT NULL default '0', PRIMARY KEY
(id_ip)
) TYPE=MyISAM;

CREATE TABLE system_arch_ip (
id_ip int(3) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', ip
bigint(20) NOT NULL default '0',
total int(11) NOT NULL default '0', PRIMARY KEY  (id_ip)
) TYPE=MyISAM;
CREATE TABLE system_arch_ip_month ( id_ip int(3) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', ip
bigint(20) NOT NULL default '0',
total int(11) NOT NULL default '0', PRIMARY KEY  (id_ip)
) TYPE=MyISAM;
CREATE TABLE system_arch_ip_week ( id_ip int(3) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', ip
bigint(20) NOT NULL default '0',
total int(11) NOT NULL default '0', PRIMARY KEY  (id_ip)
) TYPE=MyISAM;
CREATE TABLE system_arch_clients (
id_client int(11) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00',
browsers_msie int(11) NOT NULL default '0', browsers_opera int(11)
NOT NULL default '0', browsers_netscape int(11) NOT NULL default
```

```
'0', browsers_firefox INT(11) NOT NULL default '0', browsers_myie
INT(11) NOT NULL default '0', browsers_mozilla INT(11) NOT NULL
default '0', browsers_none int(11) NOT NULL default '0',
systems_windows int(11) NOT NULL default '0', systems_unix int(11)
NOT NULL default '0', systems_macintosh int(11) NOT NULL default
'0', systems_none int(11) NOT NULL default '0',
PRIMARY KEY  (id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_clients_month ( id_client int(11) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00',
browsers_msie int(11) NOT NULL default '0', browsers_opera int(11)
NOT NULL default '0',
browsers_netscape int(11) NOT NULL default '0',
browsers_firefox INT(11) NOT NULL default '0',
browsers_myie INT(11) NOT NULL default '0',
browsers_mozilla INT(11) NOT NULL default '0',
browsers_none int(11) NOT NULL default '0',
systems_windows int(11) NOT NULL default '0',
systems_unix int(11) NOT NULL default '0', systems_macintosh
int(11) NOT NULL default '0', systems_none int(11) NOT NULL default
'0', PRIMARY KEY  (id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_clients_week ( id_client int(11) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00',
browsers_msie int(11) NOT NULL default '0',
browsers_opera int(11) NOT NULL default '0', browsers_netscape
int(11) NOT NULL default '0', browsers_firefox INT(11) NOT NULL
default '0', browsers_myie INT(11) NOT NULL default '0',
browsers_mozilla INT(11) NOT NULL default '0', browsers_none
int(11) NOT NULL default '0', systems_windows int(11) NOT NULL
default '0', systems_unix int(11) NOT NULL default '0',
systems_macintosh int(11) NOT NULL default '0', systems_none
int(11) NOT NULL default '0', PRIMARY KEY  (id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_robots (
id_client int(11) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', yandex
int(11) NOT NULL default '0',
rambler int(11) NOT NULL default '0', google int(11) NOT NULL
default '0', aport int(11) NOT NULL default '0', msn int(11) NOT
NULL default '0', none int(11) NOT NULL default '0', PRIMARY KEY
(id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_robots_month ( id_client int(11) NOT NULL
auto_increment,
```

```
putdate datetime NOT NULL default '0000-00-00 00:00:00', yandex
int(11) NOT NULL default '0',
rambler int(11) NOT NULL default '0', google int(11) NOT NULL
default '0', aport int(11) NOT NULL default '0', msn int(11) NOT
NULL default '0', none int(11) NOT NULL default '0', PRIMARY KEY
(id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_robots_week ( id_client int(11) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', yandex
int(11) NOT NULL default '0',
rambler int(11) NOT NULL default '0', google int(11) NOT NULL
default '0', aport int(11) NOT NULL default '0', msn int(11) NOT
NULL default '0', none int(11) NOT NULL default '0', PRIMARY KEY
(id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_enterpoint (
id_enterpoint int(11) NOT NULL auto_increment
putdate datetime NOT NULL default '0000-00-00 00:00:00', page
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_enterpoint)
) TYPE=MyISAM;
CREATE TABLE system_arch_enterpoint_month ( id_enterpoint int(11)
NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', page
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_enterpoint)
) TYPE=MyISAM;
CREATE TABLE system_arch_enterpoint_week ( id_enterpoint int(11)
NOT NULL auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', page
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_enterpoint)
) TYPE=MyISAM;

CREATE TABLE system_arch_deep (
id_client int(11) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', visit1
int(11) NOT NULL default '0',
visit2 int(11) NOT NULL default '0', visit3 int(11) NOT NULL
default '0', visit4 int(11) NOT NULL default '0', visit5 int(11)
NOT NULL default '0', visit6 int(11) NOT NULL default '0', visit7
int(11) NOT NULL default '0', visit8 int(11) NOT NULL default '0',
visit9 int(11) NOT NULL default '0', visit10 int(11) NOT NULL
default '0', visit20 int(11) NOT NULL default '0', visit30 int(11)
NOT NULL default '0', visit40 int(11) NOT NULL default '0', visit50
int(11) NOT NULL default '0', visit60 int(11) NOT NULL default '0',
```

```
visit70 int(11) NOT NULL default '0', visit80 int(11) NOT NULL
default '0', visit90 int(11) NOT NULL default '0', visit100 int(11)
NOT NULL default '0', visitmore int(11) NOT NULL default '0',
PRIMARY KEY  (id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_deep_month ( id_client int(11) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', visit1
int(11) NOT NULL default '0',
visit2 int(11) NOT NULL default '0', visit3 int(11) NOT NULL
default '0', visit4 int(11) NOT NULL default '0', visit5 int(11)
NOT NULL default '0', visit6 int(11) NOT NULL default '0', visit7
int(11) NOT NULL default '0', visit8 int(11) NOT NULL default '0',
visit9 int(11) NOT NULL default '0', visit10 int(11) NOT NULL
default '0', visit20 int(11) NOT NULL default '0', visit30 int(11)
NOT NULL default '0', visit40 int(11) NOT NULL default '0', visit50
int(11) NOT NULL default '0', visit60 int(11) NOT NULL default '0',
visit70 int(11) NOT NULL default '0', visit80 int(11) NOT NULL
default '0', visit90 int(11) NOT NULL default '0', visit100 int(11)
NOT NULL default '0', visitmore int(11) NOT NULL default '0',
PRIMARY KEY  (id_client)
) TYPE=MyISAM;
CREATE TABLE system_arch_deep_week ( id_client int(11) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', visit1
int(11) NOT NULL default '0',
visit2 int(11) NOT NULL default '0', visit3 int(11) NOT NULL
default '0', visit4 int(11) NOT NULL default '0', visit5 int(11)
NOT NULL default '0', visit6 int(11) NOT NULL default '0', visit7
int(11) NOT NULL default '0', visit8 int(11) NOT NULL default '0',
visit9 int(11) NOT NULL default '0', visit10 int(11) NOT NULL
default '0', visit20 int(11) NOT NULL default '0', visit30 int(11)
NOT NULL default '0', visit40 int(11) NOT NULL default '0', visit50
int(11) NOT NULL default '0', visit60 int(11) NOT NULL default '0',
visit70 int(11) NOT NULL default '0', visit80 int(11) NOT NULL
default '0', visit90 int(11) NOT NULL default '0', visit100 int(11)
NOT NULL default '0', visitmore int(11) NOT NULL default '0',
PRIMARY KEY  (id_client)
) TYPE=MyISAM;

CREATE TABLE system_arch_time (
id_time int(11) NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', visit1
int(11) NOT NULL default '0',
visit2 int(11) NOT NULL default '0', visit3 int(11) NOT NULL default
'0', visit4 int(11) NOT NULL default '0', visit5 int(11) NOT NULL
default '0', visit6 int(11) NOT NULL default '0', visit7 int(11) NOT
NULL default '0', visit8 int(11) NOT NULL default '0', visit9 int(11)
```

```
NOT NULL default '0', visit10 int(11) NOT NULL default '0', visit20
int(11) NOT NULL default '0', visit30 int(11) NOT NULL default '0',
visit40 int(11) NOT NULL default '0', visit50 int(11) NOT NULL
default '0', visit60 int(11) NOT NULL default '0', visit2h int(11)
NOT NULL default '0', visit3h int(11) NOT NULL default '0', visit4h
int(11) NOT NULL default '0', visit5h int(11) NOT NULL default '0',
visit6h int(11) NOT NULL default '0', visit7h int(11) NOT NULL
default '0', visit8h int(11) NOT NULL default '0', visit9h int(11)
NOT NULL default '0', visit10h int(11) NOT NULL default '0', visit11h
int(11) NOT NULL default '0', visit12h int(11) NOT NULL default '0',
visit13h int(11) NOT NULL default '0', visit14h int(11) NOT NULL
default '0', visit15h int(11) NOT NULL default '0', visit16h int(11)
NOT NULL default '0', visit17h int(11) NOT NULL default '0', visit18h
int(11) NOT NULL default '0', visit19h int(11) NOT NULL default '0',
visit20h int(11) NOT NULL default '0', visit21h int(11) NOT NULL
default '0', visit22h int(11) NOT NULL default '0', visit23h int(11)
NOT NULL default '0', visit24h int(11) NOT NULL default '0', PRIMARY
KEY  (id_time)
) TYPE=MyISAM;
CREATE TABLE system_arch_time_month ( id_time int(11) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', visit1
int(11) NOT NULL default '0',
visit2 int(11) NOT NULL default '0', visit3 int(11) NOT NULL
default '0', visit4 int(11) NOT NULL default '0', visit5 int(11)
NOT NULL default '0', visit6 int(11) NOT NULL default '0', visit7
int(11) NOT NULL default '0', visit8 int(11) NOT NULL default '0',
visit9 int(11) NOT NULL default '0', visit10 int(11) NOT NULL
default '0', visit20 int(11) NOT NULL default '0', visit30 int(11)
NOT NULL default '0', visit40 int(11) NOT NULL default '0', visit50
int(11) NOT NULL default '0', visit60 int(11) NOT NULL default '0',
visit2h int(11) NOT NULL default '0', visit3h int(11) NOT NULL
default '0', visit4h int(11) NOT NULL default '0', visit5h int(11)
NOT NULL default '0', visit6h int(11) NOT NULL default '0', visit7h
int(11) NOT NULL default '0', visit8h int(11) NOT NULL default '0',
visit9h int(11) NOT NULL default '0', visit10h int(11) NOT NULL
default '0', visit11h int(11) NOT NULL default '0', visit12h
int(11) NOT NULL default '0', visit13h int(11) NOT NULL default
'0', visit14h int(11) NOT NULL default '0', visit15h int(11) NOT
NULL default '0', visit16h int(11) NOT NULL default '0', visit17h
int(11) NOT NULL default '0', visit18h int(11) NOT NULL default
'0', visit19h int(11) NOT NULL default '0', visit20h int(11) NOT
NULL default '0', visit21h int(11) NOT NULL default '0', visit22h
int(11) NOT NULL default '0', visit23h int(11) NOT NULL default
'0', visit24h int(11) NOT NULL default '0', PRIMARY KEY  (id_time)
) TYPE=MyISAM;
CREATE TABLE system_arch_time_temp ( time_min int(11) NOT NULL
default '0',
putdate datetime NOT NULL default '0000-00-00 00:00:00'
```

```
) TYPE=MyISAM;
CREATE TABLE system_arch_time_week ( id_time int(11) NOT NULL
auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00',
visit1 int(11) NOT NULL default '0', visit2 int(11) NOT NULL
default '0', visit3 int(11) NOT NULL default '0', visit4 int(11)
NOT NULL default '0', visit5 int(11) NOT NULL default '0', visit6
int(11) NOT NULL default '0', visit7 int(11) NOT NULL default '0',
visit8 int(11) NOT NULL default '0', visit9 int(11) NOT NULL
default '0', visit10 int(11) NOT NULL default '0', visit20 int(11)
NOT NULL default '0', visit30 int(11) NOT NULL default '0', visit40
int(11) NOT NULL default '0', visit50 int(11) NOT NULL default '0',
visit60 int(11) NOT NULL default '0', visit2h int(11) NOT NULL
default '0', visit3h int(11) NOT NULL default '0', visit4h int(11)
NOT NULL default '0', visit5h int(11) NOT NULL default '0', visit6h
int(11) NOT NULL default '0', visit7h int(11) NOT NULL default '0',
visit8h int(11) NOT NULL default '0', visit9h int(11) NOT NULL
default '0', visit10h int(11) NOT NULL default '0', visit11h
int(11) NOT NULL default '0', visit12h int(11) NOT NULL default
'0', visit13h int(11) NOT NULL default '0', visit14h int(11) NOT
NULL default '0', visit15h int(11) NOT NULL default '0', visit16h
int(11) NOT NULL default '0', visit17h int(11) NOT NULL default
'0', visit18h int(11) NOT NULL default '0', visit19h int(11) NOT
NULL default '0', visit20h int(11) NOT NULL default '0', visit21h
int(11) NOT NULL default '0', visit22h int(11) NOT NULL default
'0', visit23h int(11) NOT NULL default '0', visit24h int(11) NOT
NULL default '0', PRIMARY KEY  (id_time)
) TYPE=MyISAM;

CREATE TABLE system_arch_refferer ( id_refferer int(11) NOT NULL
auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_refferer)
) TYPE=MyISAM;
CREATE TABLE system_arch_refferer_month ( id_refferer int(11) NOT
NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_refferer)
) TYPE=MyISAM;
CREATE TABLE system_arch_refferer_week ( id_refferer int(11) NOT
NULL auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0', PRIMARY KEY  (id_refferer)
) TYPE=MyISAM;
```

```
CREATE TABLE system_arch_searchquery ( id_searchquery int(11) NOT
NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0',
searches enum('yandex','google','rambler','aport','msn') NOT NULL
default 'yandex',
PRIMARY KEY  (id_searchquery)
) TYPE=MyISAM;
CREATE TABLE system_arch_searchquery_month ( id_searchquery int(11)
NOT NULL auto_increment,
putdate datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0',
searches enum('yandex','google','rambler','aport','msn') NOT NULL
default 'yandex',
PRIMARY KEY  (id_searchquery)
) TYPE=MyISAM;
CREATE TABLE system_arch_searchquery_week ( id_searchquery int(11)
NOT NULL auto_increment,
putdate_begin datetime NOT NULL default '0000-00-00 00:00:00',
putdate_end datetime NOT NULL default '0000-00-00 00:00:00', name
tinytext NOT NULL,
total int(11) NOT NULL default '0',
searches enum('yandex','google','rambler','aport','msn') NOT NULL
default 'yandex',
PRIMARY KEY  (id_searchquery)
) TYPE=MyISAM;
CREATE TABLE system_arch_num_searchquery ( id_num_searchquery
int(11) NOT NULL auto_increment, putdate datetime NOT NULL default
'0000-00-00 00:00:00', number_yandex int(11) NOT NULL default '0',
number_google int(11) NOT NULL default '0', number_rambler int(11)
NOT NULL default '0', number_aport int(11) NOT NULL default '0',
number_msn int(11) NOT NULL default '0', number_mail int(11) NOT
NULL default '0', PRIMARY KEY  (id_num_searchquery)
) TYPE=MyISAM;
CREATE TABLE system_arch_num_searchquery_month ( id_num_searchquery
int(11) NOT NULL auto_increment, putdate datetime NOT NULL default
'0000-00-00 00:00:00', number_yandex int(11) NOT NULL default '0',
number_google int(11) NOT NULL default '0', number_rambler int(11)
NOT NULL default '0', number_aport int(11) NOT NULL default '0',
number_msn int(11) NOT NULL default '0', number_mail int(11) NOT
NULL default '0', PRIMARY KEY  (id_num_searchquery)
) TYPE=MyISAM;
CREATE TABLE system_arch_num_searchquery_week ( id_num_searchquery
int(11) NOT NULL auto_increment, putdate_begin datetime NOT NULL
default '0000-00-00 00:00:00', putdate_end datetime NOT NULL
default '0000-00-00 00:00:00', number_yandex int(11) NOT NULL
default '0',
```

```
number_google int(11) NOT NULL default '0', number_rambler int(11)
NOT NULL default '0', number_aport int(11) NOT NULL default '0',
number_msn int(11) NOT NULL default '0', number_mail int(11) NOT
NULL default '0', PRIMARY KEY  (id_num_searchquery)
) TYPE=MyISAM;
CREATE TABLE `system_cities` (
`city_id` int(11) NOT NULL default '0',
`region_id` int(11) NOT NULL default '0',
`city_name` varchar(255) NOT NULL default ''
) TYPE=MyISAM;
CREATE TABLE `system_ip_compact` (
`init_ip` bigint(20) NOT NULL default '0',
`end_ip` bigint(20) NOT NULL default '0',
`city_id` int(11) NOT NULL default '0'
) TYPE=MyISAM;
INSERT INTO `system_ip_compact` VALUES (1040547840, 1040553455, 1);
INSERT INTO `system_ip_compact` VALUES (1040553456, 1040553463, 0);
INSERT INTO `system_ip_compact` VALUES (1040553464, 1040580607, 1);
INSERT INTO `system_ip_compact` VALUES (1041244160, 1041252351,
20);
INSERT INTO `system_ip_compact` VALUES (1041252352, 1041252991, 1);
CREATE TABLE `system_regions` (
`region_id` int(11) NOT NULL default '0',
`region_name` varchar(255) NOT NULL default ''
) TYPE=MyISAM;
INSERT INTO `system_cities` VALUES (1, 77, 'Ternopil'); INSERT INTO
`system_cities` VALUES (2, 78, 'Kyiv'); INSERT INTO `system_cities`
VALUES (3, 54, 'Lviv');
INSERT INTO `system_cities` VALUES (4, 25, 'Khmelnytsky');

CREATE TABLE `system_ip_compact` (
`init_ip` bigint(20) NOT NULL default '0',
`end_ip` bigint(20) NOT NULL default '0',
`city_id` int(11) NOT NULL default '0'
) TYPE=MyISAM;
INSERT INTO `system_ip_compact` VALUES (1040547840, 1040553455, 1);
INSERT INTO `system_ip_compact` VALUES (1040553464, 1040580607, 2);
INSERT INTO `system_ip_compact` VALUES (1041252352, 1041252991, 3);
INSERT INTO `system_ip_compact` VALUES (1041253056, 1041259007, 4);
CREATE TABLE `system_regions` (
`region_id` int(11) NOT NULL default '0',
`region_name` varchar(255) NOT NULL default ''
) TYPE=MyISAM;
INSERT INTO `system_regions` VALUES (0, 'Kyiv');
INSERT INTO `system_regions` VALUES (1, 'Kyivska oblast'); INSERT
INTO `system_regions` VALUES (2, 'Ternopilska oblast'); INSERT INTO
`system_regions` VALUES (3, 'USA');
```

ДОДАТОК Б

Програмні коди сценаріївлічильника відвідувань сайту

Count.php

```
<?php
Error_Reporting(E_ALL & ~E_NOTICE);

// Встановлюємо з'єднання з базою даних
require_once("config.php");

//перевірка що шлях закінчується index.php
if(substr($_SERVER['PHP_SELF'], strlen($_SERVER['PHP_SELF']) - 1)
== "/") $_SERVER['PHP_SELF'] .= "index.php";
// Змінна для посилання на сторінку для якої проводиться підрахунок
// Формуємо стрічку з ip
if($obtip == 0) $ip = $_SERVER['REMOTE_ADDR']; // За умовчанням
if($obtip == 1) $ip = urldecode(getenv(HTTP_CLIENTIP));
if(empty($ip)) $ip = '0.0.0.0';
$  $tbl_ip             = 'system_ip';
$  $tbl_pages          = 'system_pages';
$  $tbl_refferer = 'system_refferer';
$  $tbl_searchquerys = 'system_searchquerys';
// Це стрічка з реферером - URL сторінки, з якою відвідувач прийшов
на
// сайт
$  $reff = urldecode($_SERVER['HTTP_REFERER']);
//           // З'єднуємося з сервером бази даних
$  $dbcnx = @mysql_connect($dblocation, $dbuser, $dbpasswd);
if($dbcnx)
{
// Вибираємо базу даних
if(@mysql_select_db($dbname,$dbcnx))
{
if(empty($titlepage)) $titlepage =
"http://".$_SERVER['SERVER_NAME'].$_SERVER['PHP_SELF'];
if (!get_magic_quotes_gpc()) $titlepage =
mysql_escape_string($titlepage);
$query = "SELECT id_page FROM $tbl_pages
WHERE title = '$titlepage'";
$pgs = mysql_query($query); if ($pgs)
{
// З'ясуємо, первинний ключ (id_page) поточної сторінки  (по назві
сторінки)
if(mysql_num_rows($pgs)>0) $id_page = mysql_result($pgs,
// Якщо назва цієї сторінки відсутня в таблиці pages
// те перевіряємо сторінку за її адресою.
else
{
```

```
$query = "SELECT id_page FROM $tbl_pages
WHERE
 name='".$_SERVER['PHP_SELF']."'";
$pgs = mysql_query($query); if ($pgs)
{
// З'ясуємо, первинний ключ (id_page) поточної сторінки (за
 адресою сторінки)
 if(mysql_num_rows($pgs)>0)
{

WHERE id_page=$id_page";
 $id_page = mysql_result($pgs, 0);
$query = "UPDATE $tbl_pages SET title='$titlepage'

mysql_query($query);
}

// Якщо ця сторінка відсутня в таблиці pages
// і ні разу не враховувалася - додаємо цю сторінку в таблицю.

else
{
$query = "INSERT INTO $tbl_pages VALUES (NULL,

'".$_SERVER['PHP_SELF']."','$titlepage ', 0)";
@mysql_query($query);
// З'ясовуємо первинний ключ тільки що доданою
// сторінки
$id_page = mysql_insert_id();
}
}
}
}

'mozilla';
'opera'; 'netscape'; 'firefox';
'macintosh'; 'macintosh';

// Визначаємо рядок USER_AGENT
$useragent = $_SERVER['HTTP_USER_AGENT'];
$browser = 'none';
// З'ясовуємо браузер
if(strpos($useragent, "Mozilla")!== false) $browser =
if(strpos($useragent, "MSIE") !== false) $browser = 'msie';
if(strpos($useragent, "MyIE") !== false) $browser = 'myie';
if(strpos($useragent, "Opera") !== false) $browser =
if(strpos($useragent, "Netscape")!== false) $browser =
if(strpos($useragent, "Firefox")!== false) $browser =
// З'ясовуємо операційну систему
```

```
$os = 'none';
if(strpos($useragent, "Win") !== false) $os = 'windows';
if(strpos($useragent, "Linux") !== false
|| strpos($useragent, "Lynx") !== false
|| strpos($useragent, "Unix") !== false) $os = 'unix';
if(strpos($useragent, "Macintosh")!== false) $os =
if(strpos($useragent, "PowerPC") !== false) $os =

// З'ясовуємо приналежність до пошукових роботів
if(strpos($useragent, "StackRambler")!== false) $os =
 'robot_rambler'; 'robot_google';
 if(strpos($useragent, "Googlebot") !== false) $os =
if(strpos($useragent, "Mediapartners-Google")     !== false)
 $os = 'robot_google';
if(strpos($useragent, "Yandex") !== false) $os =
 'robot_yandex';
'robot_aport'; 'robot_msnbot';
 if(strpos($useragent, "Aport") !== false) $os =
if(strpos($useragent, "msnbot") !== false) $os =
 $search = 'none';
// З'ясовуємо приналежність до пошукових систем
if(strpos($reff,"yandex")) $search = 'yandex';
if(strpos($reff,"rambler")) $search = 'rambler';
if(strpos($reff,"google")) $search = 'google';
if(strpos($reff,"aport"))     $search = 'aport';
= 'mail';
= 'msn';
 if(strpos($reff,"mail") && strpos($reff,"search"))    $search
if(strpos($reff,"msn") && strpos($reff,"results"))     $search
$server_name = $_SERVER["SERVER_NAME"];
if(substr($_SERVER["SERVER_NAME"],0,4) == "www.")
$server_name = substr($_SERVER["SERVER_NAME"],4);
if(strpos($reff,$server_name)) $search = 'own_site';

// Заносимо усю зібрану інформацію в базу даних
$query_main = "INSERT INTO $tbl_ip VALUES (
NULL INET_ATON('$ip') NOW()
$id_page
'    '$browser
'    '$os')";
@mysql_query($query_main);
// Якщо є реферер, заносимо інформацію про нього в окрему таблицю
if(!empty($reff) && $search=="none")
{
$reff = str_replace("'","'",$reff);
$query_reff = "INSERT INTO $tbl_refferer VALUES ( NULL
'    '$reff
now() INET_ATON('$ip')
$id_page)"; @mysql_query($query_reff);
```

```
}
//вноситься пошуковий запит у відповідну таблицю
if(!empty($reff) && $search!="none" && $search != "own_site")
{
switch($search)
{
case 'yandex ':
{
eregi("text=([^&]*) ", $reff."&", $query);
if(strpos($reff,"yandpage")!=null)
$quer=convert_cyr_string(urldecode($query[1]),
else
$quer=$query[1]; break;
}
 case 'rambler ':
{
eregi("words=([^&]*) ", $reff."&", $query);
$quer = $query[1]; break;
}
case 'mail ':
{
eregi("q=([^&]*) ", $reff."&", $query);
$quer = $query[1]; break;
}
case 'google ':
{
eregi("q=([^&]*) ", $reff."&", $query);
$quer = utf8_win($query[1]); break;
}
case 'msn ':

{
eregi("q=([^&]*) ", $reff."&", $query);
$quer = utf8_win($query[1]); break;
}
case 'aport ':
{
eregi("r=([^&]*) ", $reff."&", $query);
$quer = $query[1]; break;");
 }
}//кінець для switch
$symbols = array("\"", "'", "(", ")", "+", ",", "-

$quer = str_replace($symbols, " ", $quer);
$quer = trim($quer);
$quer = preg_replace('|[\s]+|',' ',$quer);
$sql="INSERT INTO $tbl_searchquerys VALUES (NULL, '$quer ',
 now(), INET_ATON('$ip'), $id_page, '$search')";
```

```
@mysql_query($sql);
}
}
}
function utf8_win($s)
{
$s=str_replace("\xD0\xB0","а",$s);
$s=str_replace("\xD0\x90","А",$s);
$s=str_replace("\xD0\xB1","б",$s);
$s=str_replace("\xD0\x91","Б",$s);
$s=str_replace("\xD0\xB2","в",$s);
$s=str_replace("\xD0\x92","В",$s);
$s=str_replace("\xD0\xB3","г",$s);
$s=str_replace("\xD0\x93","Г",$s);
$s=str_replace("\xD0\xB4","д",$s);
$s=str_replace("\xD0\x94","Д",$s);
$s=str_replace("\xD0\xB5","е",$s);
$s=str_replace("\xD0\x95","Е",$s);
$s=str_replace("\xD1\x91","е",$s);
$s=str_replace("\xD0\x81","Е",$s);
$s=str_replace("\xD0\xB6","ж",$s);
$s=str_replace("\xD0\x96","Ж",$s);
$s=str_replace("\xD0\xB7","з",$s);
$s=str_replace("\xD0\x97","З",$s);
$s=str_replace("\xD0\xB8","і",$s);
$s=str_replace("\xD0\x98","І",$s);
$s=str_replace("\xD0\xB9","й",$s);
$s=str_replace("\xD0\x99","Й",$s);
$s=str_replace("\xD0\xBA","к",$s);
$s=str_replace("\xD0\x9A","К",$s);
$s=str_replace("\xD0\xBB","л",$s);
$s=str_replace("\xD0\x9B","Л",$s);
$s=str_replace("\xD0\xBC","м",$s);
$s=str_replace("\xD0\x9C","М",$s);
$s=str_replace("\xD0\xBD","н",$s);
$s=str_replace("\xD0\x9D","Н",$s);
$s=str_replace("\xD0\xBE","о",$s);
$s=str_replace("\xD0\x9E","О",$s);
$s=str_replace("\xD0\xBF","п",$s);
$s=str_replace("\xD0\x9F","П",$s);
$s=str_replace("\xD1\x80","р",$s);
$s=str_replace("\xD0\xA0","Р",$s);
$s=str_replace("\xD1\x81","з",$s);
$s=str_replace("\xD0\xA1","З",$s);
$s=str_replace("\xD1\x82","т",$s);
$s=str_replace("\xD0\xA2","Т",$s);
$s=str_replace("\xD1\x83","у",$s);
$s=str_replace("\xD0\xA3","У",$s);
```

```
$s=str_replace("\xD1\x84","ф",$s);
$s=str_replace("\xD0\xA4","Ф",$s);
$s=str_replace("\xD1\x85","х",$s);
$s=str_replace("\xD0\xA5","Х",$s);
$s=str_replace("\xD1\x86","ц",$s);
$s=str_replace("\xD0\xA6","Ц",$s);
$s=str_replace("\xD1\x87","ч",$s);
$s=str_replace("\xD0\xA7","Ч",$s);
$s=str_replace("\xD1\x88","ш",$s);
$s=str_replace("\xD0\xA8","Ш",$s);
$s=str_replace("\xD1\x89","щ",$s);
$s=str_replace("\xD0\xA9","Щ",$s);
$s=str_replace("\xD1\x8A","ъ",$s);
$s=str_replace("\xD0\xAA","Ъ",$s);
$s=str_replace("\xD1\x8B","ы",$s);
$s=str_replace("\xD0\xAB","Ы",$s);
$s=str_replace("\xD1\x8C","ь",$s);
$s=str_replace("\xD0\xAC","Ь",$s);
$s=str_replace("\xD1\x8D","э",$s);
$s=str_replace("\xD0\xAD","Э",$s);
$s=str_replace("\xD1\x8E","ю",$s);
$s=str_replace("\xD0\xAE","Ю",$s);
$s=str_replace("\xD1\x8F","я",$s);
$s=str_replace("\xD0\xAF","Я",$s);
return $s;
}
?>
```

Config.php

```
<?php

// Основні змінні
// Ім'я сервера бази даних, наприклад
// $dblocation = "mysql 28.noweb.ru"
// зараз виставлений сервер локальної машини
$dblocation = "localhost";
//   // Ім'я бази даних, на хостингу або локальній машині
$dbname = "count";
// Ім'я користувача бази даних
$dbuser = "root";
// і його пароль
$dbpasswd = "pGf8E34";
// Спосіб визначення IP -адреса відвідувача
// 0 Підходить для більшості хостингів у тому числі
// для використання на локальній машині
$obtip = 0;
// 1 На деяких хостингах ip -адрес відвідувача не
// заноситься в змінну $REMOTE_ADDR, до них відноситься
// наприклад www.nodex.ru
```

```
// $obtip = 1;
// Поточна версія системи
if($version=@file("version.txt"))
$version = $version[0]; else $version = 0;
?>

Archive.php
<?php
set_time_limit(0);

Error_Reporting(E_ALL & ~E_NOTICE);
// Встановлюємо з'єднання з базою даних
require_once("config.php");
// Бібліотека функцій архівації require_once("utils_hits.php");
require_once("utils_ip.php"); require_once("utils_client.php");
require_once("utils_robots.php");
require_once("utils_enterpoints.php");
require_once("utils_deep.php"); require_once("utils_time.php");
require_once("utils_refferer.php");
require_once("utils_num_search.php");
require_once("utils_search.php");
// Отримуємо дату початку реєстрації даних і число що пройшли з
початку реєстрації
$dat = mysql_query("SELECT UNIX_TIMESTAMP(MIN(putdate)) AS data
FROM
$tbl_ip");
if (!$dat) exit(mysql_error());
$arr_date = mysql_fetch_array($dat);
// Останій повний день
$last_day = mktime(23,59,59, date("m"), date("d") - 1, date("Y"));
// Отримуємо останню дату архівації
$query = "SELECT UNIX_TIMESTAMP(MAX(putdate)) FROM $tbl_arch_hits";
$arh = mysql_query($query); if(mysql_num_rows($arh)> 0)
{
$last_date = mysql_result($arh, 0);
}
if(empty($last_date))
{
if(isset($arr_date['data']))
{
// Якщо запуск перший - беремо дату з $tbl_searchquerys
$last_date = $arr_date['data'];

}
else
{
// Інакше беремо поточну добу
$last_date = time();
}
```

```
}
//блок архівації
if (ceil(($last_day - $last_date)/24/60/60))
{
////////////////////////////////////////////////
// Архівуємо інформацію в щоденні таблиці
//////////////////////////////////////////////// archive_ip
     ($tbl_ip, $tbl_arch_ip); archive_client ($tbl_ip,
$tbl_arch_clients); archive_robots ($tbl_ip, $tbl_arch_robots);
archive_enterpoints ($tbl_ip, $tbl_pages, $tbl_arch_enterpoint);
archive_deep   ($tbl_ip, $tbl_arch_deep);
archive_time   ($tbl_ip, $tbl_arch_time, $tbl_arch_time_temp);
archive_refferer    ($tbl_refferer, $tbl_arch_refferer);
archive_searchquery ($tbl_searchquerys, $tbl_arch_searchquery);
archive_num_searchquery ($tbl_searchquerys,
$tbl_arch_num_searchquery); archive_hit_hosts     ($tbl_ip,
$tbl_arch_hits);
////////////////////////////////////////////////
// Видаляємо старі записи
////////////////////////////////////////////////
$query = "SELECT MAX(putdate) FROM $tbl_arch_hits";
$arh = mysql_query($query);
if(!$arh) exit("Збій при видаленні старих записів");
if(mysql_num_rows($arh)> 0)
{
$last_date_arch = mysql_result($arh, 0);
$arr[] = "DELETE FROM $tbl_ip WHERE putdate <= '$last_date_arch' -
INTERVAL 30 DAY";
$arr[] = "DELETE FROM $tbl_refferer WHERE putdate <=
'$last_date_arch' - INTERVAL 30 DAY";
$arr[] = "DELETE FROM $tbl_searchquerys WHERE putdate <=
'$last_date_arch' - INTERVAL 30 DAY";
foreach($arr as $query) if(!mysql_query($query))
exit(mysql_error());
}
////////////////////////////////////////////////
// Архівуємо інформацію в щотижневі таблиці
////////////////////////////////////////////////
archive_hit_hosts_week   ($tbl_arch_hits, $tbl_arch_hits_week);
archive_ip_week     ($tbl_arch_ip, $tbl_arch_ip_week);
archive_client_week ($tbl_arch_clients,
$tbl_arch_clients_week);
archive_robots_week ($tbl_arch_robots, $tbl_arch_robots_week);
archive_enterpoints_week ($tbl_arch_enterpoint,
$tbl_arch_enterpoint_week);
archive_deep_week   ($tbl_arch_deep, $tbl_arch_deep_week);
archive_time_week   ($tbl_arch_time, $tbl_arch_time_week);
archive_refferer_week   ($tbl_arch_refferer,
$tbl_arch_refferer_week);
```

```
archive_searchquery_week ($tbl_arch_searchquery,
$tbl_arch_searchquery_week);
archive_num_searchquery_week ($tbl_arch_num_searchquery,
$tbl_arch_num_searchquery_week);

//////////////////////////////////////////////////
// Архівуємо інформацію в еженемесячные таблиці
//////////////////////////////////////////////////
archive_hit_hosts_month  ($tbl_arch_hits, $tbl_arch_hits_month);

archive_ip_month    ($tbl_arch_ip, $tbl_arch_ip_month);
archive_clients_month    ($tbl_arch_clients,
$tbl_arch_clients_month);
archive_robots_month     ($tbl_arch_robots,
$tbl_arch_robots_month);
archive_enterpoints_month     ($tbl_arch_enterpoint,
$tbl_arch_enterpoint_month);
archive_deep_month  ($tbl_arch_deep, $tbl_arch_deep_month);
archive_time_month  ($tbl_arch_time, $tbl_arch_time_month);
archive_refferer_month   ($tbl_arch_refferer,
$tbl_arch_refferer_month);
archive_searchquery_month     ($tbl_arch_searchquery,
$tbl_arch_searchquery_month);
archive_num_searchquery_month ($tbl_arch_num_searchquery,
$tbl_arch_num_searchquery_month);
}
?>
```