МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2023_ р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

АВТОМАТИЗОВАНЕ ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.1910218

Виконав студент 4-го курсу, групи 402
_____ Б. О. Пархоменко
«___» _____ 2023_ р
Керівник: д-р. пед. наук, проф.
_____ О. П. Мещанінов
«___» _____ 2023_ р

Миколаїв – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р. техн. наук, проф.
_____ Ю. П. Кондратенко

«____» _____2022 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Видано студенту групи 402 факультету комп'ютерних наук Пархоменко Богдану Ігоровичу.

1. Тема кваліфікаційної роботи «Автоматизоване дослідження інформаційних систем».

Керівник роботи Мещанінов Олександр Павлович, д-р пед. наук, професор.

Затв. наказом Ректора ЧНУ ім. Петра Могили від «___» _____ 20___ р. № ____

2. Строк представлення кваліфікаційної роботи студентом «___» _____ 20__ р.

3. Вхідні (початкові) дані до роботи: архітектура інформаційної системи, перелік функцій розроблюваньої системи, дані користувачів сайту.

Очікуваний результат: аналіз отриманих даних, відображення результатів роботи системи дослідження інформаційних систем.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

- аналіз області інформаційних систем;
- огляд існуючих аналогів розроблювальної системи;

- визначення сучасних методів збору, обробки та збереження даних;
- визначення сучасних технологій для аналізу та візуалізації даних;
- проектування та розробка системи дослідження інформаційних систем.

5. Перелік графічного матеріалу: загальна схема інформаційних систем, архітектура системи дослідження ІС, блок-схема алгоритму роботи системи , UML діаграма прецедентів, презентація.

6. Завдання до спеціальної частини: «Оцінка умов праці для працівника, аналіз факторів виробничого середовища у приміщенні»

7. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	Боженко А.Л. викладач	

Керівник роботи д-р. пед. наук, проф. Мещанінов О.П. _____
(наук. ступінь, вчене звання, прізвище та ініціали) (підпис)

Завдання прийнято до виконання __Пархоменко Б.І.___ _____
(прізвище та ініціали) (підпис)

Дата видачі завдання «__» _____ ____ 20__ р.

КАЛЕНДАРНИЙ ПЛАН
виконання бакалаврської кваліфікаційної роботи

Тема: Автоматизоване дослідження інформаційних систем

	Найменування роботи	**Початок**	**Закінчення**	**Примітки**
	Подання заяви на затвердження теми та керівників БКР	30.10.2022	30.10.2022	Виконано
	Отримання завдання на виконання БКР	25.11.2022	25.11.2022	Виконано
	Складання календарного плану роботи на весь період виконання БКР	10.12.2022	10.12.2022	Виконано
	Отримання завдання на переддипломну практику	01.05.2023	01.05.2023	Виконано
	Проходження переддипломної практики, збір та аналіз матеріалів до БКР	01.05.2023	14.05.2022	Виконано
	Розробка звіту з переддипломної практики	15.05.2023	17.05.2023	Виконано
	Аналіз предметної області, постановка задачі та формування технічного завдання	04.04.2023	15.04.2023	Виконано
	Огляд та визначення основних методів та алгоритмів для реалізації функціоналу	19.05.2023	24.05.2023	Виконано
0	Моделювання та прототипна реалізація системи	02.06.2023	15.06.2023	Виконано
1	Попередній захист БКР на засіданні комісії кафедри	30.05.2023	30.05.2023	Виконано
2	Доробка та остаточне оформлення БКР	02.06.2023	19.06.2023	Виконано
3	Подання БКР рецензенту	15.06.2023	17.06.2023	Виконано
4	Подання БКР, її електронної копії та інших документів (відгуку, рецензії) до захисту	19.06.2023	22.06.2023	Виконано
5	Захист БКР перед екзаменаційною комісією (ЕК)	26.06.2023	29.06.2023	

Розробив студент __Пархоменко Б.І._____ _____
(прізвище, ім'я, по батькові студента)
(*підпис*)
Керівник роботи __ д-р пед. наук, проф. Мещанінов О. П. _____
(*посада, прізвище, ім'я, по батькові)* *(підпис)*
«_ _» _____ 2023 р.

АНОТАЦІЯ

бакалаврської кваліфікаційної роботи студента групи 402 ЧНУ ім. Петра Могили

Пархоменка Богдана Ігоровича

Тема: «Автоматизоване дослідження інформаційних систем»

Актуальність роботи полягає в потребі оптимізації інформаційних систем, адже автоматизоване дослідження інформаційних систем дозволить виявити слабкі місця, ідентифікувати проблеми та запропонувати шляхи вдосконалення.

Об’єкт роботи — процеси дослідження інформаційних систем.

Предмет роботи — методи збору, обробки, зберігання даних, а також аналізу та візуалізації.

Метою бакалаврської роботи є автоматизація дослідження інформаційних систем з використанням методів збору даних про відвідування сайту та аналізу поведінки користувачів (загальний час перебування на сайті, найпопулярніша сторінка серед користувачів, heatmap та ін).

Пояснювальна записка до кваліфікаційної роботи складається зі вступу, трьох розділів, висновків та додатків.

У першому розділі проводиться аналіз предметної сфери, виконується огляд та аналіз систем, що мають схожі функції та вирішують такі ж задачі, що поставленні перед нашою розроблювальною системою.

У другому розділі розглянуто різні методи та технології для реалізації функціоналу системи. Запропоновано декілька методів та алгоритмів для реалізації основного функціоналу серед яких виділено основі, котрі будуть використовуватися при розробці системи.

У третьому розділі розроблено блок-схему детального процесу роботи автоматизованої системи. Створено UML діаграму прецедентів, де відображена взаємодія користувача з прецедентами автоматизованої системи. Розроблено прототип автоматизованої системи дослідження інформаційних систем та реалізовано окремі головні функції системи.

В результаті розроблено прототип системи дослідження інформаційних систем.

Бакалаврська кваліфікаційна робота містить 56 сторінок, 22 рисунків, 18 використаних джерел та 4 додатки.

Ключові слова: інформаційні системи, аналіз даних, моделювання систем, аналітика даних, автоматизація, база даних.

ABSTRACT

Bachelor's qualification work of a student of the 402 group of the Petro Mohyla National University

Parkhomenko Bogdan

Automated research of information systems

The relevance of the work lies in the need to optimize information systems, because automated research of information systems will identify weaknesses, identify problems and suggest ways to improve.

Object of work - a program for the study of information systems.

The subject of the work is the methods of collecting, processing, storing data, as well as analysis and visualization.

The purpose of the bachelor's thesis is to automate the research of information systems using methods of data collection on site visits and analysis of user behavior (total time spent on the site, the most popular page among users, heatmap, etc.).

Explanatory note to the qualification work consists of an introduction, three sections, conclusions and appendices.

In the first section, an analysis of the subject area is carried out, an overview and analysis of systems that have similar functions and solve the same tasks as set before our development system is carried out.

The second section discusses various methods and technologies for implementing the functionality of the system. Several methods and algorithms for the implementation of the main functionality are proposed, among which the basis that will be used in the development of the system is allocated.

In the third section, a block diagram of the detailed process of the automated system is developed. A UML diagram of precedents was created, where the interaction of the user with the precedents of the automated system is displayed. A prototype of an automated information systems research system was developed and some main functions of the system were implemented.

As a result, a prototype of the information systems research system was developed.

The bachelor qualification paper contains 56 pages, 22 figures, 18 sources used and 4 annexes.

Keywords: information systems, data analysis, systems modeling, data analytics, automation, database.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

API - Application Programming Interface

URL - Uniform Resource Locator

БД - база даних

ІС - інформаційні системи

ПЗ - програмне забезпечення

ВСТУП

Сучасне суспільство стикається зі зростаючим обсягом інформації та розмаїттям інформаційних систем, що використовуються в різних галузях. Це ставить перед науковими дослідниками і фахівцями завдання розуміння, аналізу та оптимізації цих систем. Автоматизовані методи дослідження інформаційних систем стають невід'ємною складовою процесу розвитку та управління інформаційними технологіями.

Одним з ключових значень інформаційних систем є полегшення доступу до інформації. Завдяки ним, ми маємо можливість швидко знаходити потрібну нам інформацію з різних джерел, таких як Інтернет, бази даних, електронні бібліотеки та інші. Це розширює наші знання, допомагає нам вирішувати проблеми та приймати обґрунтовані рішення.

Крім того, інформаційні системи сприяють розвитку науки та досліджень. Вони дозволяють збирати, аналізувати та обробляти великі обсяги даних, виявляти закономірності, проводити моделювання та прогнозування.

Метою бакалаврської роботи є автоматизація дослідження інформаційних систем з використанням методів збору даних про відвідування сайту та аналізу поведінки користувачів (загальний час перебування на сайті, найпопулярніша сторінка серед користувачів, heatmap та ін).

Об’єкт роботи – процеси дослідження інформаційних систем.

Предмет роботи – методи збору, обробки, збереження даних, а також аналізу та візуалізації.

Актуальність роботи полягає в потребі оптимізації інформаційних систем, адже автоматизоване дослідження інформаційних систем дозволить виявити слабкі місця, ідентифікувати проблеми та запропонувати шляхи вдосконалення.

Основні **завдання,** які повинні бути розв’язані в ході даної роботи:

- аналіз області інформаційних систем;
- огляд існуючих аналогів розроблювальної системи;
- визначення сучасних методів збору, обробки та збереження даних;

– визначення сучасних технологій для аналізу та візуалізації даних;
– проектування та розробка системи дослідження інформаційних систем.

1 АНАЛІЗ ПРЕДМЕТНОЇ СФЕРИ. ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд на предметну сферу

Більшість людей котрі чують термін інформаційна система, думають що це якесь програмне забезпечення, яке створене для зберігання інформації. Назва дійсно вводить в оману, однак інформаційна система це дещо більше.

Інформаційна система — це поєднання програмного забезпечення, апаратного забезпечення та телекомунікаційних мереж для збору корисних даних, особливо в організації. Її використовують для зберігання, обробки та видачі інформації з метою вирішення конкретного завдання. Такі системи допомагають багатьом підприємствам завершити певні задачі, керувати операціями та взаємодіяти зі своїми споживачами на випередження конкурентів. Багато відомих компаній такі як, Amazon, Google та eBay повністю побудовані на інформаційних технологіях.

Основне завдання інформаційної системи збирати інформацію, що необхідна підприємствам або організаціям, для налагодження управління усіма внутрішніми процесами, керування своїми ресурсами. Щоб робота інформаційної системи була успішною необхідно дотримуватися наступних пунктів у її створенні [1]:

– визначення інформаційних потреб;
– набір джерел інформації;
– збір інформації;
– введення інформації із зовнішніх або внутрішніх джерел;
– обробка інформації;
– вивід інформації споживачам, або передача її у іншу систему;
– передача зібраної інформації для розподілення серед поставлених задач (розробка прогнозів, оцінка тенденцій, розробка стратегії та ін.);
– забезпечення зворотного зв’язку інформації опрацьованої людьми та редагування вхідної інформації.

В основі роботи інформаційної системи лежить процес створення інформації. Також, ІС, по суті, складається з п’яти основних компонентів, що саме і виконують

здійснення введення, обробки, виведення, зворотного зв’язку та контролю інформації: обладнання, програмного забезпечення, бази даних, мережі та людей (див.рис.1.1).

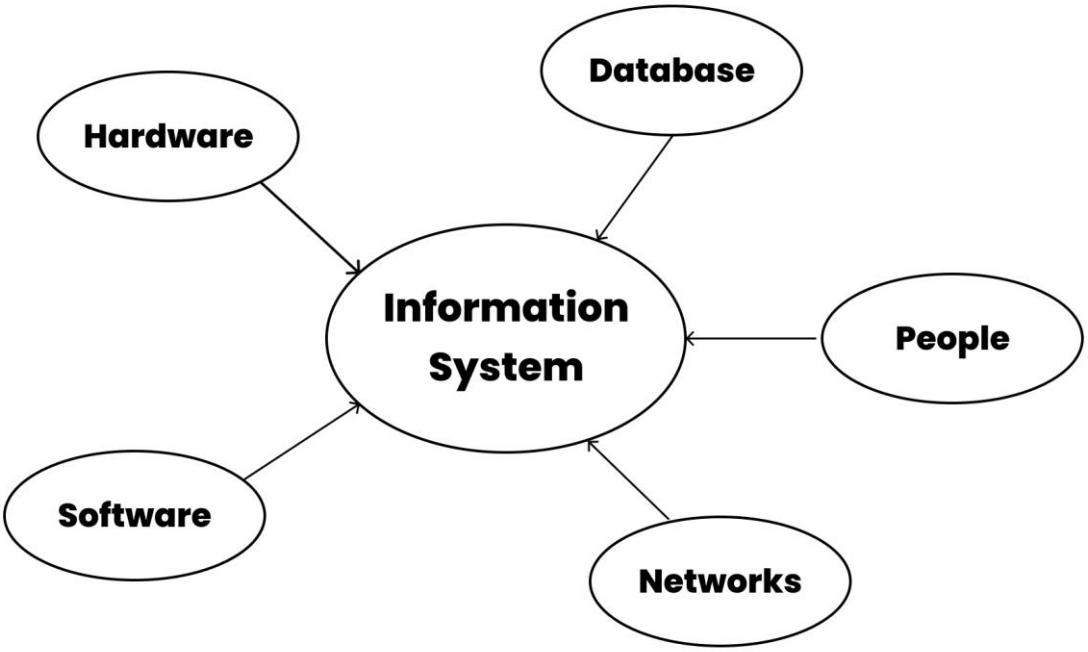

Рисунок 1.1 – Загальна схема ІС

Обладнання (Hardware) — це комп’ютери, смартфони, планшети та навіть смарт-годинники, загалом будь-який девайс, що має змогу збирати, зберігати, отримувати доступ і керувати величезними обсягами даних.

Програмне забезпечення (Software) — його можна розділити на два типи: системне ПЗ, яке дозволяє користувачу керувати файлами комп’ютера та загальним інтерфейсом (прикладом такого ПЗ можуть бути операційні системи); прикладне ПЗ, програми, що виконують певні завдання. Системне ПЗ створює початкову точку, з якої може будуватися прикладне програмне забезпечення [2].

База даних (Database) — це сховища, що зберігають якісну і кількісну інформацію, яку програмне забезпечення та користувачі отримують, обробляють і аналізують відповідно до своїх потреб.

Мережі (Networks) — це телекомунікаційні мережі, такі як Інтернет. На сьогоднішній день, вони дуже необхідні для успішної роботи всіх типів організацій і їх комп'ютерних інформаційних систем. Телекомунікаційні мережі складаються з комп'ютерів, комунікаційних процесорів та інших пристроїв, котрі з'єднані між собою засобами зв'язку та керовані комунікаційним ПЗ [3]. Концепція мережевих ресурсів полягає в тому, що комунікаційні мережі це основний ресурсний компонент будь-яких інформаційних систем.

Люди (Peoples) — це оператори, адміністратори та інші системні спеціалісти.

Усі продукти інформаційних технологій стали частиною нашого повсякденного життя.

По-перше, інформаційні системи стали необхідними для розвитку будь-якого бізнесу. Кожне підприємство проводить певні комп'ютерні операції, які необхідні для виконання їх роботи. Багато організацій потребують комп'ютерних програмних забезпечень, реалізація мережевої архітектури для досягнення цілей бізнесу або розробки програм, веб-сайтів чи ігор. Отже, будь-яка компанія, яка прагне забезпечити своє майбутнє, потребує інтеграції добре розробленої інформаційної системи.

По-друге, ІС забезпечує краще зберігання та доступ до даних. Введення даних вручну займають значно багато часу, ніж це роблять інформаційні системи.

По-третє, ІС допомагає бізнесу в процесі прийняття рішень. Завдяки інформаційній системі надання всієї важливої інформації легше для прийняття кращих рішень. Також, інформаційна система дозволяє співробітникам ефективно спілкуватися. Так як документи зберігаються в папках, співробітникам легше ділитися ними та отримувати до них доступ [4].

1.2 Огляд на аналіз наявних аналогів

Системи дослідження інформаційних систем використовуються для вивчення та аналізу різних аспектів інформаційних систем. Ось кілька прикладів таких систем:

– система моніторингу трафіку (Network Traffic Monitoring System) — ця система дозволяє вивчати трафік, що протікає через комп'ютерну мережу. Вона реєструє та аналізує дані про вхідний та вихідний трафік, ідентифікує загрози та надає детальну статистику про використання ресурсів мережі;

– система моніторингу баз даних (Database Monitoring System) — ця система дозволяє спостерігати за роботою баз даних і аналізувати їх продуктивність та ефективність. Вона надає інформацію про запити до бази даних, виявляє проблеми з продуктивністю та надає рекомендації щодо оптимізації роботи бази даних;

– система аудиту інформаційної безпеки (Information Security Audit System) — ця система допомагає визначити потенційні загрози та слабкі місця в інформаційній системі. Вона проводить аудит системи, перевіряє виконання безпекових політик та стандартів, виявляє вразливості та рекомендує заходи для забезпечення безпеки даних.

Метою бакалаврської роботи є дослідження, проектування та розробка веб-застосунку, що буде збирати статистику користувача на сайті, а саме загальний час перебування на сайті, найпопулярніша сторінка серед користувачів, heatmap. Тому нижче розглянемо приклади аналогів таких систем.

Google Analytics є однією з найпопулярніших систем аналітики в Інтернеті. Вона надає розширені засоби збору та аналізу даних про користувачів, такі як час на сторінці, статистика переходів, поведінка користувачів та багато іншого. Вона дозволяє створювати звіти, візуалізувати дані та отримувати цінні інсайти щодо ефективності вашого сайту (див.рис.1.2).

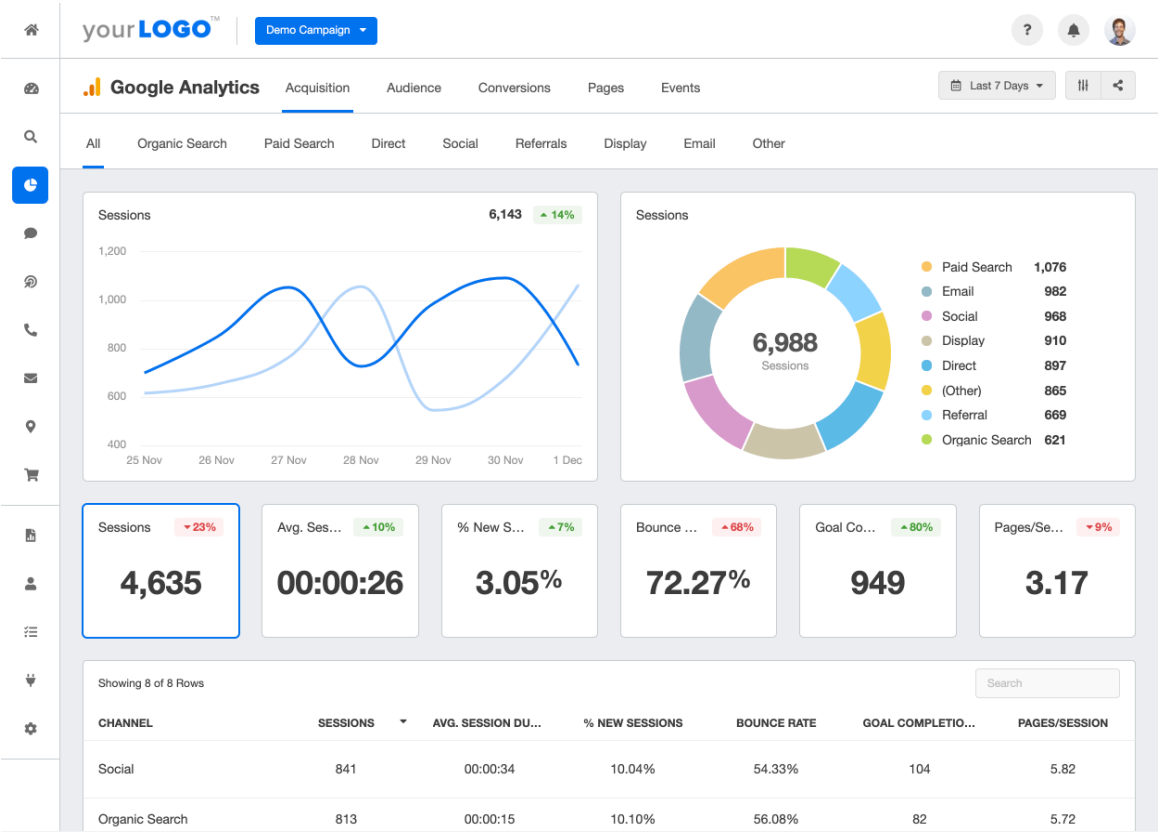

Рисунок 1.2 – Google Analytics Dashboard

Вона надає широкий набір інструментів для збору, аналізу та відстеження даних про веб-сторінки, що дозволяє вам зрозуміти, як користувачі взаємодіють з вашим сайтом. Основні функції Google Analytics включають:

– *відстеження відвідувань.* Google Analytics збирає дані про кількість відвідувачів, відвідувань, сторінок, переходів, перебування на сторінці та інші метрики, що дозволяють вам отримати загальну картину про активність користувачів на вашому сайті;

– *джерела трафіку.* Ви можете визначити, з яких джерел (наприклад, пошукових систем, соціальних мереж, рекламних кампаній) приходить трафік на ваш сайт. Це допомагає вам оцінити ефективність різних маркетингових каналів та прийняти стратегічні рішення щодо просування вашого сайту;

– *аналіз поведінки користувачів.* Google Analytics дозволяє вам розуміти, як користувачі взаємодіють зі сторінками вашого сайту. Ви можете відстежувати час, проведений на сторінці, шляхи, які користувачі проходять по вашому сайту, та

багато іншого. Це дає вам можливість виявити слабкі місця та вдосконалити користувацький досвід;

– *цілі та конверсії.* Ви можете налаштувати цілі в Google Analytics, щоб відстежувати конверсії, такі як реєстрація на сайті, заповнення форми або покупка товару. Це дозволяє вам вимірювати ефективність вашого сайту та маркетингових кампаній.

Yandex.Metrica є аналогом Google Analytics, розробленим Яндексом. Вона надає схожий набір функцій, включаючи статистику переходів, відстеження цілей, аналіз публіки, аналіз поведінки користувачів та інші корисні функції.

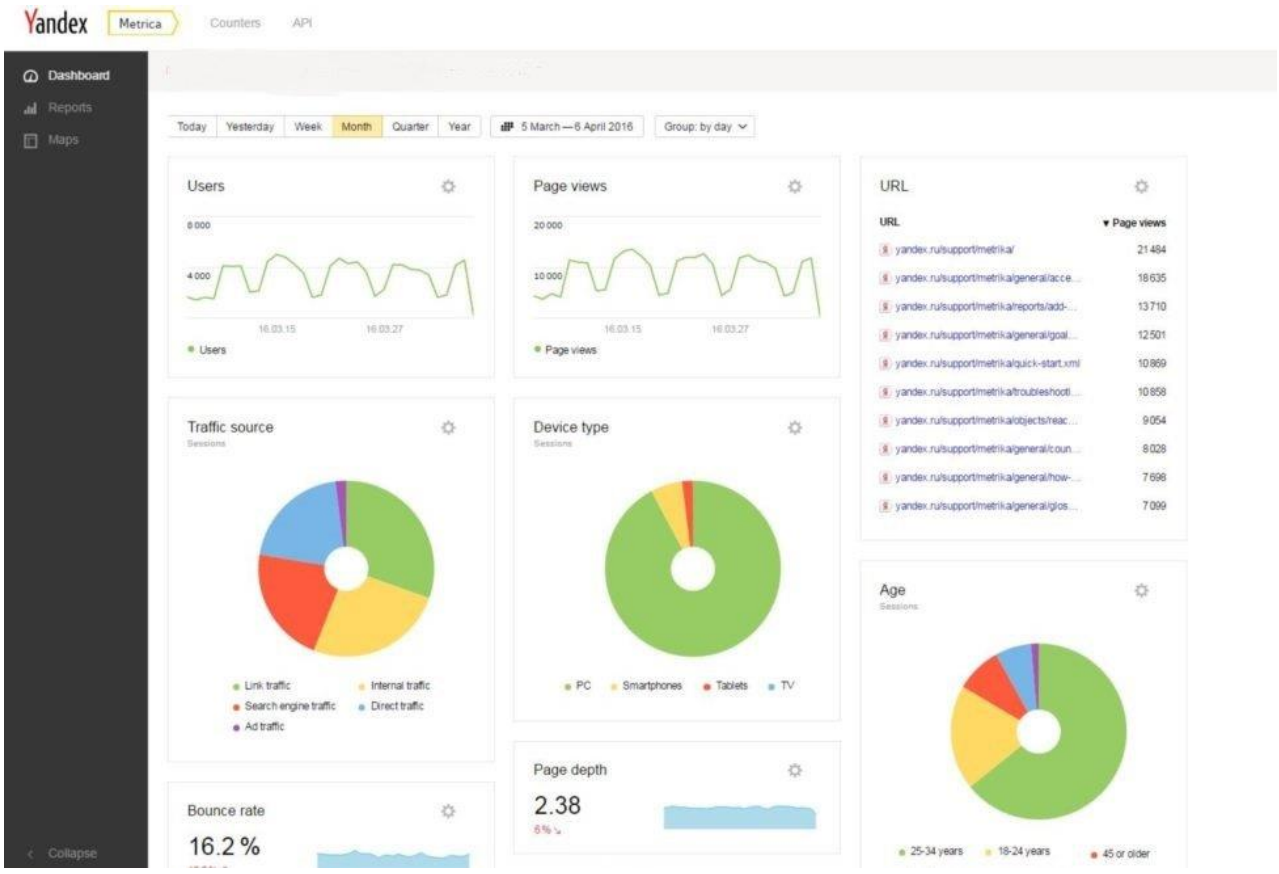

Рисунок 1.3 – Yandex Metrica Dashboard

Основні функції Yandex.Metrica включають:

– *збір даних про відвідування.* Yandex.Metrica відстежує кількість відвідувачів, відвідувань сторінок, переходів та інші метрики, що дозволяють вам отримати загальне уявлення про активність користувачів на вашому сайті;

– *аналіз джерел трафіку.* Ви можете визначити, з яких джерел (пошукових систем, соціальних мереж, рекламних кампаній) приходить трафік на ваш сайт. Це допомагає вам зрозуміти, які маркетингові канали працюють найкраще для вашого сайту;

– *аналіз поведінки користувачів*. Yandex.Metrica надає детальну інформацію про поведінку користувачів на вашому сайті. Ви можете відстежувати час, проведений на сторінці, кількість прокрутки сторінки, взаємодію з елементами сайту та інше. Це дозволяє виявити сильні та слабкі сторони вашого сайту та покращити його користувацький досвід;

– *відстеження цілей та конверсій.* Ви можете налаштувати цілі в Yandex.Metrica для відстеження конверсій, таких як замовлення товарів, реєстрація на сайті або завантаження файлів. Ви отримуєте дані про конверсії та вимірюєте ефективність вашого сайту;

– *теплові карти та відеозапис екранів користувачів.* Yandex.Metrica надає можливість створювати теплові карти, які візуалізують області сторінок, на які користувачі найбільше звертають увагу. Це допомагає виявити гарячі та холодні зони на вашому сайті і зрозуміти, як користувачі сприймають його. Крім того, ви можете записувати відеозаписи сеансів користувачів, щоб переглянути, як вони взаємодіють з вашим сайтом в реальному часі.

Hotjar є інструментом веб-аналітики, який забезпечує збір даних про поведінку користувачів на веб-сайті. Він надає можливість записувати сеанси користувачів, створювати теплові карти натискань, аналізувати форми та багато іншого. Це дозволяє збирати цінну інформацію про те, як користувачі взаємодіють з вашим сайтом (рис. 1.4).

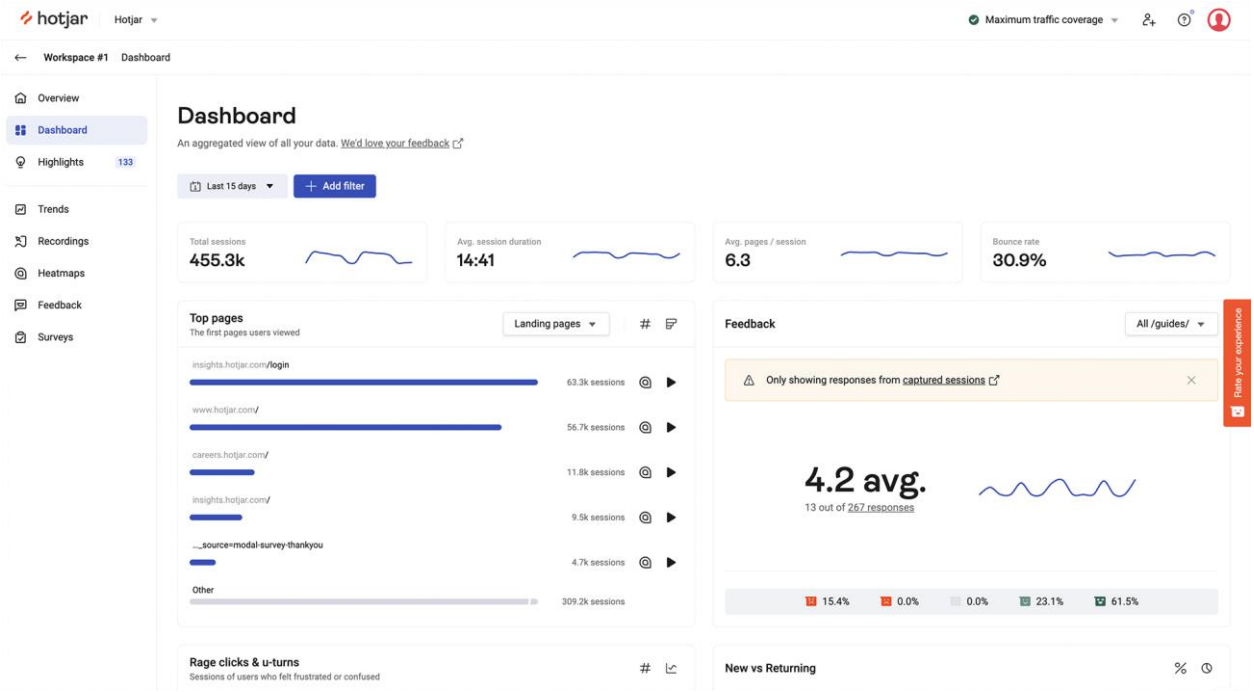

Рисунок 1.4 – Hotjar Dashboard

Основні функції Hotjar включають:

– *теплові карти (Heatmaps).* Hotjar дозволяє вам створювати теплові карти, які візуалізують області сторінок, на які користувачі найбільше звертають увагу. Ви можете побачити, де саме користувачі клікають, прокручують сторінку або наводять курсор миші. Це допомагає виявити гарячі та холодні зони на вашому сайті та зрозуміти, як користувачі сприймають його;

– *запис сеансів користувачів.* Ви можете записувати відеозаписи сеансів користувачів, що дозволяє переглянути, як вони взаємодіють з вашим сайтом. Це надає вам можливість побачити реальний процес переходу користувачів між сторінками, взаємодії з елементами і виявити можливі проблеми або перешкоди, з якими вони зіштовхуються;

– *опитування та форми зворотного зв'язку.* Hotjar дозволяє створювати опитування та форми зворотного зв'язку, які можна розміщувати на вашому сайті. Ви можете отримати важливі відгуки та інсайти від користувачів щодо їхнього досвіду, причин відхилень і проблем, а також зібрати демографічну інформацію про аудиторію.

– *воронка конверсій (Conversion Funnels).* Hotjar дозволяє створювати воронки конверсій, що візуалізують шляхи користувачів від початкової сторінки до досягнення конкретної цілі. Ви можете відстежувати, на якому етапі користувачі покидають сайт або не досягають поставленої мети, що допомагає виявити проблемні місця та вдосконалювати їх для збільшення конверсій.

1.3 Постановка задачі та формування технічного завдання

Порівнявши огляд на предметну область та проаналізувавши існуючі аналоги розроблюваної системи, можна перейти до етапу формування функціональних вимог до системи дослідження інформаційних систем. Для початку визначимо основні задачі, які буде виконувати наша розроблювальна система:

– авторизація користувача (можливість реєстрації нового);

– збір даних про відвідування (кількість відвідувачів, відвідування сторінок);

– аналіз поведінки користувачів (найвідвідуваніша сторінка, час перебування на сторінці);

– теплові карти (область сайту на яку користувачі найбільше звертають увагу);

– візуалізація отриманих даних.

В нашій розроблювальній системі дослідження інформаційних систем основним джерелом вхідної інформації є сам користувач, завдяки діям якого система буде відстежувати, обробляти та зберігати необхідні дані для закриття основних задач.

Архітектура системи дослідження інформаційних систем може варіюватись залежно від конкретних вимог та потреб дослідження. Проте, загальна архітектура такої системи може включати наступні компоненти: збір даних, зберігання даних, обробка даних, аналітика та візуалізація. Нижче представлена схема нашої майбутньої системи дослідження інформаційних систем (див.рис.1.5)

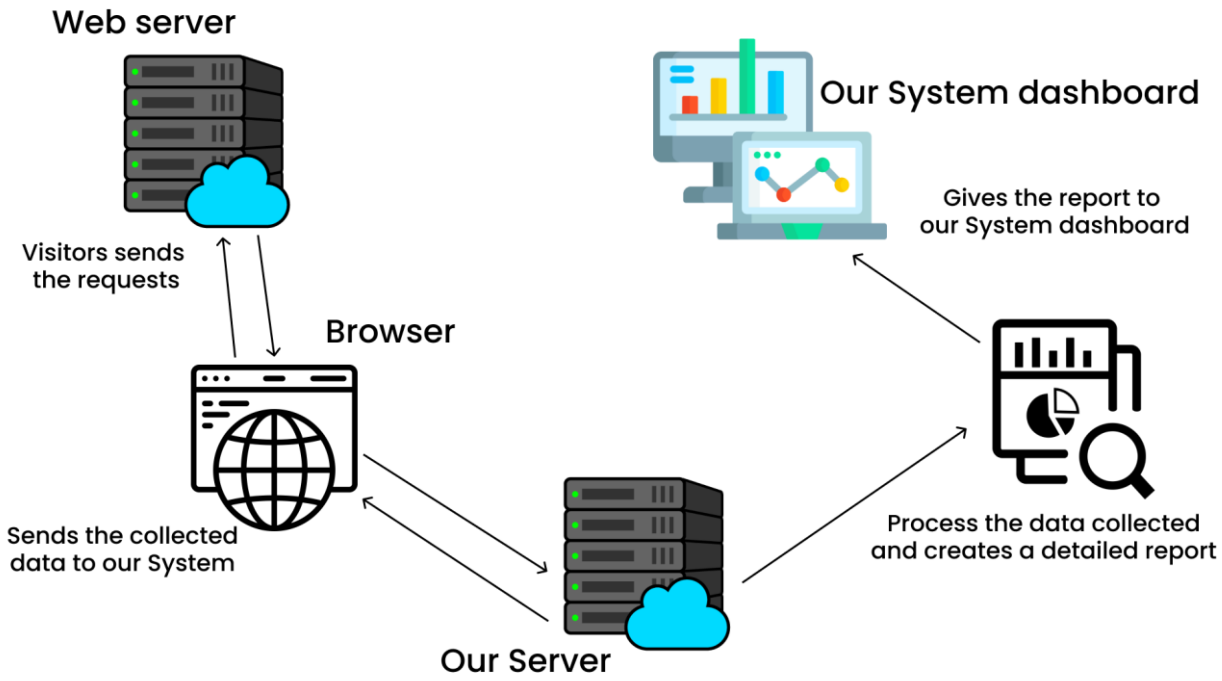

Рисунок 1.5 – Архітектура системи дослідження ІС

Система дослідження інформаційних систем повинна задовольняти певні вимоги, які можна поділити на функціональні та нефункціональні. Ось деякі загальні вимоги до такої системи.

Функціональні вимоги:

– *збір даних.* Система повинна мати можливість збирати дані про поведінку користувачів, включаючи сторінки, які вони відвідують, події, які вони виконують та інші важливі метрики;

– *аналітика даних.* Система повинна надавати функціонал для обробки та аналізу зібраних даних, включаючи статистичний аналіз, сегментацію користувачів, знаходження кореляцій та виявлення патернів;

– *візуалізація даних.* Система повинна забезпечувати можливість візуалізації даних у зручній формі, такі як графіки, діаграми, таблиці, теплові карти тощо.

Нефункціональні вимоги:

– *надійність.* Система повинна бути надійною, забезпечуючи стабільну роботу і збереження даних навіть у випадку відмови окремих компонентів;

– *швидкодія.* Система повинна працювати ефективно та забезпечувати швидкий доступ до даних та результатів аналізу;

– *масштабованість.* Система повинна бути здатною масштабуватися, щоб обробляти зростаючий об’єм даних;

– *конфіденційність та безпека.* Система повинна забезпечувати захист конфіденційності та цілісності зібраних даних. Це може включати механізми шифрування, контролю доступу, анонімізації даних та захист від несанкціонованого доступу;

– *зручність використання.* Система повинна бути зручною для використання та надавати інтуїтивно зрозумілий інтерфейс користувача. Це допомагає забезпечити ефективну роботу зі зібраними даними та аналітичними функціями;

– *підтримка.* Система повинна мати підтримку та можливість отримання допомоги у разі виникнення проблем або запитань. Це може включати документацію, навчальні матеріали, форуми або технічну підтримку.

Ці вимоги можуть бути більш конкретизовані та доповнені в залежності від конкретного дослідження інформаційних систем та вимог бізнесу. Розробка такої системи вимагає детального аналізу потреб і вимог користувачів, а також врахування технічних обмежень та можливостей.

Висновки до 1 розділу

Отже в першому розділі проведено огляд предметної сфери, визначено основне поняття ІС, основні процеси, що лежать в роботі інформаційної системи. Також, визначили, що ІС складається з п’яти основних компонентів, що саме і виконують здійснення введення, обробки, виведення, зворотного зв’язку та

контролю інформації: обладнання, програмного забезпечення, бази даних, мережі та людей.

Виконали огляд та провели аналіз систем, що мають схожі функції та вирішують такі ж задачі, що поставленні перед нашою розроблювальною системою. Детально описали основний функціонал обраних аналогів.

Визначили основні задачі, що буде виконувати наша система, створили її архітектуру та поставили основі функціональні та нефункціональні вимоги при створенні та роботі автоматизованої системи дослідження інформаційних систем.

2 ТЕХНОЛОГІЇ ТА МЕТОДИ ДЛЯ РЕАЛІЗАЦІЇ ФУНКЦІОНАЛУ СИСТЕМИ ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Для реалізації функціоналу системи дослідження інформаційних систем можуть бути використані різні технології та методи.

Технології веб-скрапінгу, API-інтеграції можна використовувати для автоматичного збору даних з різних джерел, починаючи від веб-сторінки, закінчуючи соціальними мережами. Наступним кроком зібрані дані можуть бути оброблені за допомогою аналітичних методів, статичного аналізу, машинного навчання та інших методів.

Для дослідження різних аспектів інформаційних систем можуть застосовуватися методи моделювання та симуляції. Це дозволяє відтворити реальні процеси, розглянути різні сценарії та визначити їх вплив на систему. Моделі можуть бути побудовані за допомогою математичних методів, агентних моделей, системної динаміки тощо.

Нижче розглянемо основні алгоритми та методи для реалізації функціоналу розроблювальної системи.

2.1 Алгоритм PageRank

Алгоритм PageRank або алгоритм Google був введений Ларрі Пейджем, одним із засновників Google. Спочатку він був використаний для ранжування веб-сторінок у пошуковому двигуні Google. В даний час його все більше і більше використовують у багатьох різних галузях, наприклад, для ранжування користувачів у соціальних медіа та ін. Цікавим у алгоритмі PageRank є те, як розпочати зі складної проблеми та отримати дуже просте рішення. Використаємо ранжування веб-сторінок як випадок використання для ілюстрації алгоритму PageRank.

Веб-сайт можна представити у вигляді спрямованого графа, де вузли представляють веб-сторінки, а ребра утворюють посилання між ними. Зазвичай,

якщо вузол (веб-сторінка) *i* має посилання на вузол *j*, це означає, що *i* посилається на *j* [5] (див.рис.2.1).

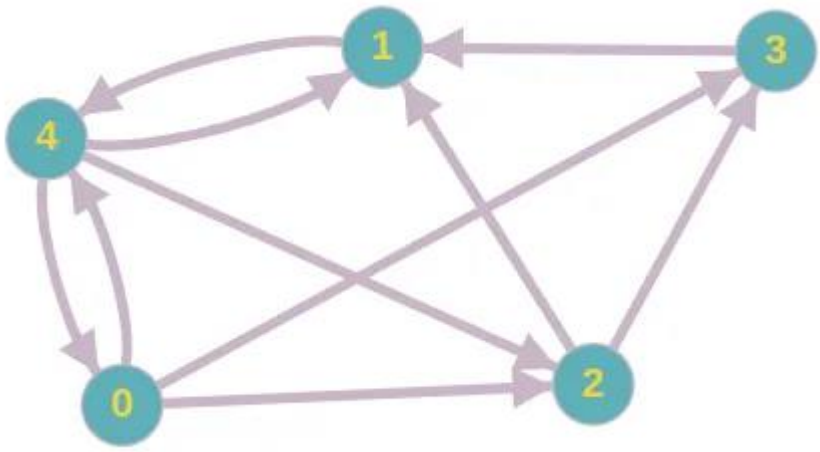

Рисунок 2.1 – Приклад орієнтованого графа

Ми повинні визначити, що таке важливість веб-сторінки. Перш за все, ми можемо сказати, що це загальна кількість веб-сторінок, які посилаються на неї. Якщо ми зупинимося на цьому критерії, то важливість цих веб-сторінок, які на неї посилаються, не береться до уваги. Іншими словами, важлива веб-сторінка і менш важлива мають однакову вагу. Інший підхід полягає в тому, що веб-сторінка розподіляє свою важливість рівномірно на всі веб-сторінки, на які вона посилається. Зробивши це, ми можемо визначити оцінку вузла *j* за формулою 2.1:

$$r_j = \sum_{i \to j} \frac{r_i}{d_i} \tag{2.1}$$

де r_i — оцінка вузла *i*, а d_i — його вихідний ступінь.

З наведеного вище прикладу ми можемо записати цю лінійну систему формула 2.2:

$$\begin{cases} r_0 = \frac{r_4}{3} \\ r_1 = \frac{r_2}{2} + \frac{r_4}{3} + r_3 \\ r_2 = \frac{r_0}{3} + \frac{r_4}{3} \\ r_3 = \frac{r_2}{2} + \frac{r_0}{3} \\ r_4 = \frac{r_0}{3} + r_1 \end{cases} \quad (2.2)$$

Передаючи праву частину цієї лінійної системи в ліву, ми отримуємо нову. Але це рішення обмежене для малих графів. Дійсно, оскільки цей вид графів розріджений, і усунення Гауса змінює матрицю при виконанні її операцій, ми втрачаємо розрідженість матриці, і це займе більше простору пам'яті. У гіршому випадку матрицю більше не можна зберігати.

Основна ідея алгоритму PageRank полягає в тому, щоб надати вагу сторінкам на основі кількості посилань, що спрямовують на ці сторінки, а також важливості сторінок, які надають ці посилання. Чим більше посилань вказує на певну сторінку, тим більша її важливість. Однак, вага посилань залежить також від важливості сторінок, які надають ці посилання. Якщо сторінка з високим рангом посилає посилання на іншу сторінку, то ця сторінка отримує більшу вагу.

Алгоритм PageRank можна уявити як процес розподілу "голосів" або ваги між веб-сторінками. Початково кожна сторінка має однакову вагу. Потім вага розподіляється між сторінками, які на неї посилаються, з урахуванням їхньої важливості. Цей процес повторюється ітеративно, поки не досягнутий стабільний розподіл ваги між сторінками.

Алгоритм PageRank має велике значення для пошукових систем, оскільки він допомагає визначити релевантність та ранг сторінок під час пошуку.

Алгоритм PageRank продовжується шляхом ітеративного розподілу ваги між веб-сторінками досягнення стабільного стану. Кожна ітерація алгоритму включає такі кроки:

Крок 1. Ініціалізація ваги: Початкова вага кожної сторінки може бути рівною значенню 1 або будь-якому іншому початковому значенню.

Крок 2. Розподіл ваги: Кожна сторінка розподіляє свою вагу між сторінками, на які вона посилається. Це робиться шляхом поділу ваги на кількість посилань, що виходять з даної сторінки. Чим більше посилань має сторінка, тим меншу частку своєї ваги вона передає кожному посиланню.

Крок 3. Акумуляція ваги: Сторінки, які отримують посилання від інших сторінок, накопичують вагу, яку вони отримують від посилань. Це робиться шляхом сумування ваги, що передається від посилань.

Крок 4. Перерозподіл ваги: Отримана вага перерозподіляється між усіма сторінками з урахуванням їхньої важливості. Цей процес повторюється декілька разів або досягнення стабільності ваги.

Крок 5. Визначення рангів сторінок: Після досягнення стабільного стану ваги, ранги сторінок можуть бути визначені на основі отриманої ваги. Сторінки з більшою вагою вважаються більш важливими та мають вищий ранг.

Значення алгоритму PageRank полягає в тому, що воно дозволяє визначити релевантність сторінок на основі їхньої важливості, визначеної через посилання між сторінками (див. Додаток А).

2.2 Cookie Algorithm

Одна із головних функцій розроблювальної системи дослідження інформаційних систем це визначення кількості відвідувачів сайту. Для визначення кількості відвідувачів сайту можна використовувати наступні алгоритми:

– ***алгоритм лічильника (Counter Algorithm)***. Це досить простий алгоритм лічильник якого збільшується кожного разу, коли сторінка сайту завантажується.

Просте збільшення лічильника дозволяє визначити загальну кількість відвідувачів, але не забезпечує інформацію про унікальних відвідувачів;

– ***алгоритм IP-адреси (IP Address Algorithm)***. Цей алгоритм використовує унікальні IP-адреси відвідувачів для визначення їхньої кількості. Він зберігає список IP-адрес, які вже відвідали сайт, і перевіряє, чи нова IP-адреса належить до списку. Це дозволяє визначити кількість унікальних відвідувачів, але може бути недостатньо точним, оскільки кілька відвідувачів можуть використовувати одну IP-адресу (наприклад, у випадку використання спільного Інтернет-підключення);

– ***алгоритм кукі-файлів (Cookie Algorithm).*** Цей алгоритм використовує унікальні ідентифікатори, які зберігаються в кукі-файлах веб-браузерів відвідувачів. Коли відвідувач завантажує сторінку сайту, сервер перевіряє наявність та унікальність цього ідентифікатора. Це дозволяє визначити кількість унікальних відвідувачів, але може бути обмеженою точністю, оскільки відвідувачі можуть блокувати або видаляти кукі-файли.

Із трьох вищеперерахованих кращим буде Cookie Algorithm. Алгоритм використовує HTTP-кукі, які зберігаються на браузері відвідувача і передаються при кожному запиті до сервера. За допомогою цих кукі можна відстежувати унікальних відвідувачів та визначити їх кількість.

Алгоритм Cookie є поширеним і ефективним способом визначення кількості відвідувачів. Однак, варто зазначити, що цей алгоритм може бути обмежений, якщо відвідувачі блокують куки або використовують приватний режим браузера, що призводить до недостовірності інформації. Реалізація алгоритму Cookie:

```
// Отримання значення кукі
function getCookie(name) {
  const cookieValue = document.cookie.match(`(^|;)\\s*${name}\\s*=\\s*([^;]+)`);
  return cookieValue ? cookieValue.pop() : '';
}

// Встановлення кукі
function setCookie(name, value, days) {
  const expirationDate = new Date();
```

```
  expirationDate.setTime(expirationDate.getTime() + (days * 24 * 60 * 60 * 1000));
  const expires = `expires=${expirationDate.toUTCString()}`;
  document.cookie = `${name}=${value}; ${expires}; path=/`;
}

// Перевірка наявності кукі
function checkCookie() {
  const cookieName = 'visitor';
  const visitorId = getCookie(cookieName);

  if (visitorId !== '') {
    // Кукі вже існує, відвідувач не рахується як новий
    console.log('Visitor already exists');
  } else {
    // Кукі не існує, встановлюємо новий кукі
    const newVisitorId = generateUniqueId(); // Генеруємо унікальний ідентифікатор
    setCookie(cookieName, newVisitorId, 365); // Встановлюємо кукі на 365 днів
    console.log('New visitor created');
  }
}

// Генерування унікального ідентифікатора
function generateUniqueId() {
  // Реалізація генерації унікального ідентифікатора за потреби
  // Можна використовувати, наприклад, timestamp або рандомні значення
  return 'unique_id';
}

// Виклик функції перевірки кукі
checkCookie();
```

У цьому коді ‘*getCookie’* використовується для отримання значення кукі за назвою. ‘*setCookie’* встановлює новий кукі з унікальним ідентифікатором, якщо кукі не існує. ‘*checkCookie’* перевіряє наявність кукі і викликає ‘*generateUniqueId’* для генерації нового ідентифікатора в разі потреби.

Це лише простий приклад, і реалізація може варіюватись в залежності від потреб вашого веб-сайту та використовуваної технології.

2.3 Реалізація теплових карт на веб-сайті

Теплові карти (Heatmaps) – це візуальний спосіб представлення даних, який демонструє розподіл чи інтенсивність певного явища, значень або дій на основі кольорової шкали. Такі карти дозволяють швидко візуалізувати густину, зміни або розподіл певних даних на площині.

У веб-розробці теплові карти використовуються для аналізу поведінки користувачів на веб-сайті. Вони відображають, де саме користувачі найбільше фокусують свою увагу на сторінці, як вони прокручують сторінку, взаємодіють з елементами, натискують на посилання тощо. Теплові карти можуть допомогти виявити недосяжні або непривабливі елементи на сторінці, виявити місця з високим рівнем зацікавленості користувачів та вдосконалити дизайн та структуру сторінки для поліпшення користувацького досвіду.

Часто їх використовують для виявлення кліків та натискань на веб-сторінці, де кожен елемент сторінки має свій колір відповідно до його популярності. Найбільш натискані або торкані елементи можуть мати яскраво-червоний колір, тоді як менш натискані елементи плавно переходять до холодних кольорів (див.рис.2.2) [6].

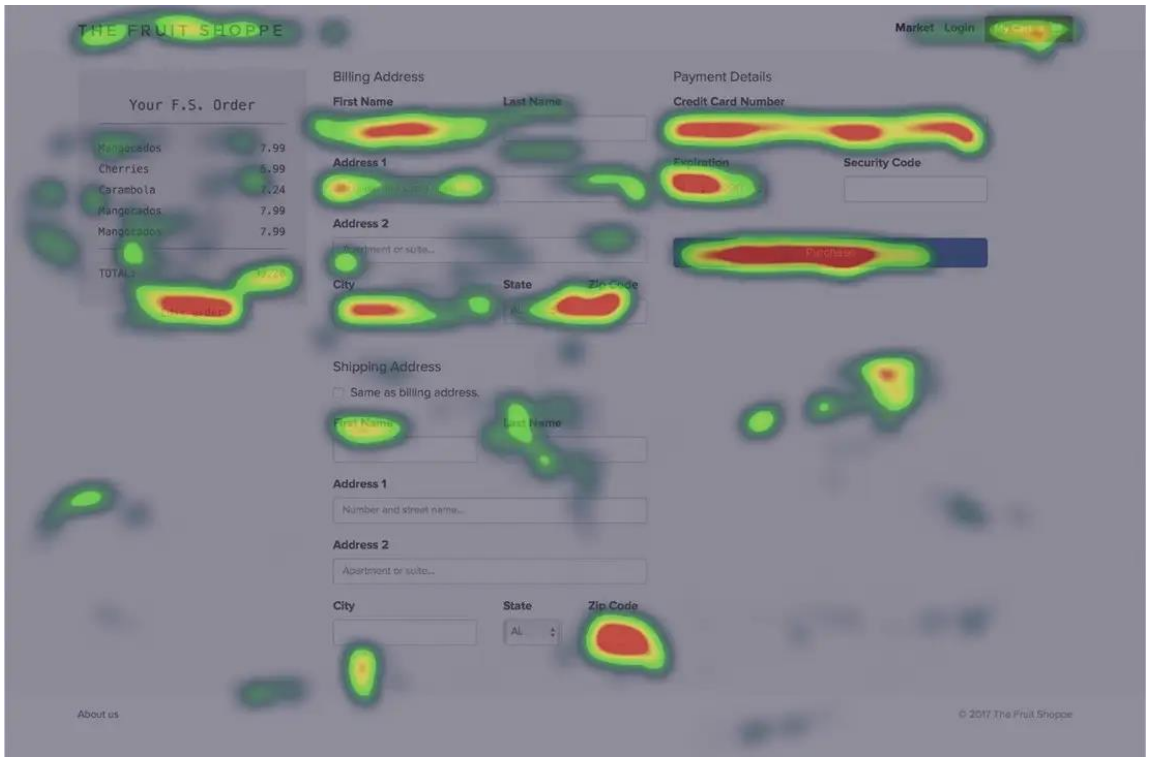

Рисунок 2.2 – Приклад теплової карти взаємодій користувачів на сторінці оформлення замовлення.

Для реалізації теплових карт на веб-сайті можна використовувати наступні підходи та алгоритми: JavaScript-бібліотеки, Google Maps API, Canvas або SVG, Сторонні сервіси та ін. Для розробки нашої системи використання JavaScript-бібліотек буде зручним рішенням.

Існують багато готових JavaScript-бібліотек, які допоможуть вам створити теплові карти на веб-сайті. Деякі популярні бібліотеки включають Heatmap.js, Leaflet.heat та D3.js. Ці бібліотеки надають можливості для візуалізації даних у вигляді теплових карт з використанням різних алгоритмів, таких як kernel density estimation або grid-based approaches.

JavaScript-бібліотека Heatmap.js дозволяє створювати теплові карти на основі вхідних даних. Ця бібліотека проста у використанні та надає різні можливості для візуалізації даних у вигляді теплової карти [7]. Функціональність Heatmap.js може бути налаштована та розширена залежно від ваших потреб. Наприклад, ви можете змінити кольорову палітру, налаштувати значення прозорості, додати підписи осей

та легенду, а також використовувати різні типи градієнтів для візуалізації даних (див. Додаток Б).

2.4 Реляційні бази даних

Реляційні бази даних (Relational databases) - це один з найпоширеніших типів баз даних, які використовуються для організації та збереження даних. У реляційній моделі дані зберігаються у вигляді таблиць зі стовпцями і рядками, де кожен рядок представляє запис, а стовпці - поля або атрибути.

Серед реляційних БД можна виділити PostgreSQL, MySQL, Oracle.

Реляційна база даних на основі PostgreSQL це одна з найпопулярніших відкритих реляційних баз даних, яка надає потужні можливості для зберігання, управління і опрацювання структурованих даних. Ця база даних підтримує запити як SQL, так і JSON.

Також, вона має вбудовану підтримку географічних даних і може обробляти складні операції, такі як розширені запити і аналітичні функції. PostgreSQL має високий рівень надійності, забезпечуючи цілісність даних і механізми відновлення після відмови. Вона також пропонує різні методи шифрування даних. Вона може працювати з великими обсягами даних і підтримує розподілені обчислення. БД може бути розгорнута на кластері серверів для розподіленого зберігання даних і обробки завдань паралельно. PostgreSQL дозволяє розширяти її функціональність за допомогою розширень і власних типів даних. Вона підтримує розробку власних функцій і модулів на різних мовах програмування. Одна з головних переваг реляційної бази даних це потужна мова запитів SQL, яка дозволяє виконувати складні запити до бази даних. Вона також має оптимізатор запитів, який старається забезпечити найкращу продуктивність виконання запитів.

PostgreSQL є безкоштовним і доступним для використання в комерційних проектах. Вона підтримується на багатьох операційних системах і може бути інтегрована з різними програмними інструментами і мовами програмування.

MySQL Database є системою клієнт-сервер, яка складається з багатопотокового SQL-сервера, який підтримує різні бекенди, кілька різних клієнтських програм і бібліотек, адміністративні інструменти та широкий спектр програмних інтерфейсів (API). Ми також надаємо MySQL як вбудовану багатопотокову бібліотеку, яку можна посилити на вашу програму, щоб отримати менший, швидший, легше керований самостійний продукт [8].

MySQL швидка, надійна, масштабована та легка у використанні. Ця система була розроблена для швидкої обробки великих баз даних і використовувалася в дуже вимогливих виробничих середовищах протягом багатьох років.

Ця система має багато переваг, по-перше, це легкість у використанні, навіть розробник-початківець здатен встановити MySQL за лічені хвилини і зможе з легкістю нею керувати. По-друге, надійність, вже протягом 25 років вона була протестована в різних сценаріях великих потужних компаній по всьому світу. Прикладами таких компаній досі виступають Facebook, Twitter, YouTube, Yahoo. По-третє, MySQL може масштабуватися, щоб задовольнити потреби найбільш відвідуваних додатків. Вбудована архітектура реплікації MySQL дозволяє організаціям, таким як Facebook, масштабувати додатки для підтримки мільярдів користувачів. Тобто, ця система не стране перешкодою, у випадку якщо ви будете розширювати можливості свого застосунку або розроблювальної системи.

MySQL є відкритою системою, доступною безкоштовно для використання, і має широку підтримку та активне співтовариство розробників. Вона може бути інтегрована з різними веб-серверами і програмними мовами, включаючи PHP, Python, Java та інші.

Тому при створення системи дослідження інформаційних систем, MySQL може буде хорошим вибором для зберігання даних і виконання запитів, оскільки вона надає потужні можливості для аналізу та опрацювання структурованих даних, легка у використанні, надійна та має максимальну гнучкість у розробці.

Висновки до 2 розділу

Отже в другому розділі розглянуто різні методи та технології для реалізації функціоналу системи дослідження інформаційних систем. Запропоновано декілька методів та алгоритмів для реалізації основного функціоналу серед яких виділили основі, котрі будуть використовуватися при розробці нашої системи.

Визначили, що для реалізації функції релевантності сторінок, найкраще підійде алгоритм PageRank, адже це досить легкий та дієвий алгоритм. Алгоритм кукі-файлів (Cookie Algorithm), допоможе нам визначити кількість відвідувачів сайту, серед запропонованих у розділі алгоритмів, він впорається найкраще.

Також, для відтворення теплових карт, було вирішено використовувати JavaScript бібліотеки. А в якості бази даних було обрано реляційну базу даних MySQL.

3 МОДЕЛЮВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

3.1 Алгоритм роботи ігрового застосунку.

Мета нашої розроблювальної системи – це веб-застосунок, що буде збирати дані про відвідування сайту та аналізувати поведінку користувачів (загальний час перебування на сайті, найпопулярніша сторінка серед користувачів, heatmap та ін)..

Розроблювальна система призначена для швидкого збору та аналізу даних про відвідувачів. Основні функціональні можливості системи:

– можливість авторизуватися (реєстрація);
– збір даних про відвідування;
– аналіз поведінки користувачів;
– теплові карти;
– візуалізація отриманих даних.

Блок-схема допоможе схематично представити процес роботи автоматизованої системи дослідження інформаційних систем (рис. 3.1).

Перший крок. Ініціалізація користувача.

Другий крок. Якщо користувач зареєстрований вводимо дані і входимо в систему, в іншому випадку проходимо реєстрацію.

Третій крок. Вхід у кабінет користувача.

Четвертий крок. Перегляд виведених даних.

П’ятий крок. Вихід із кабінету.

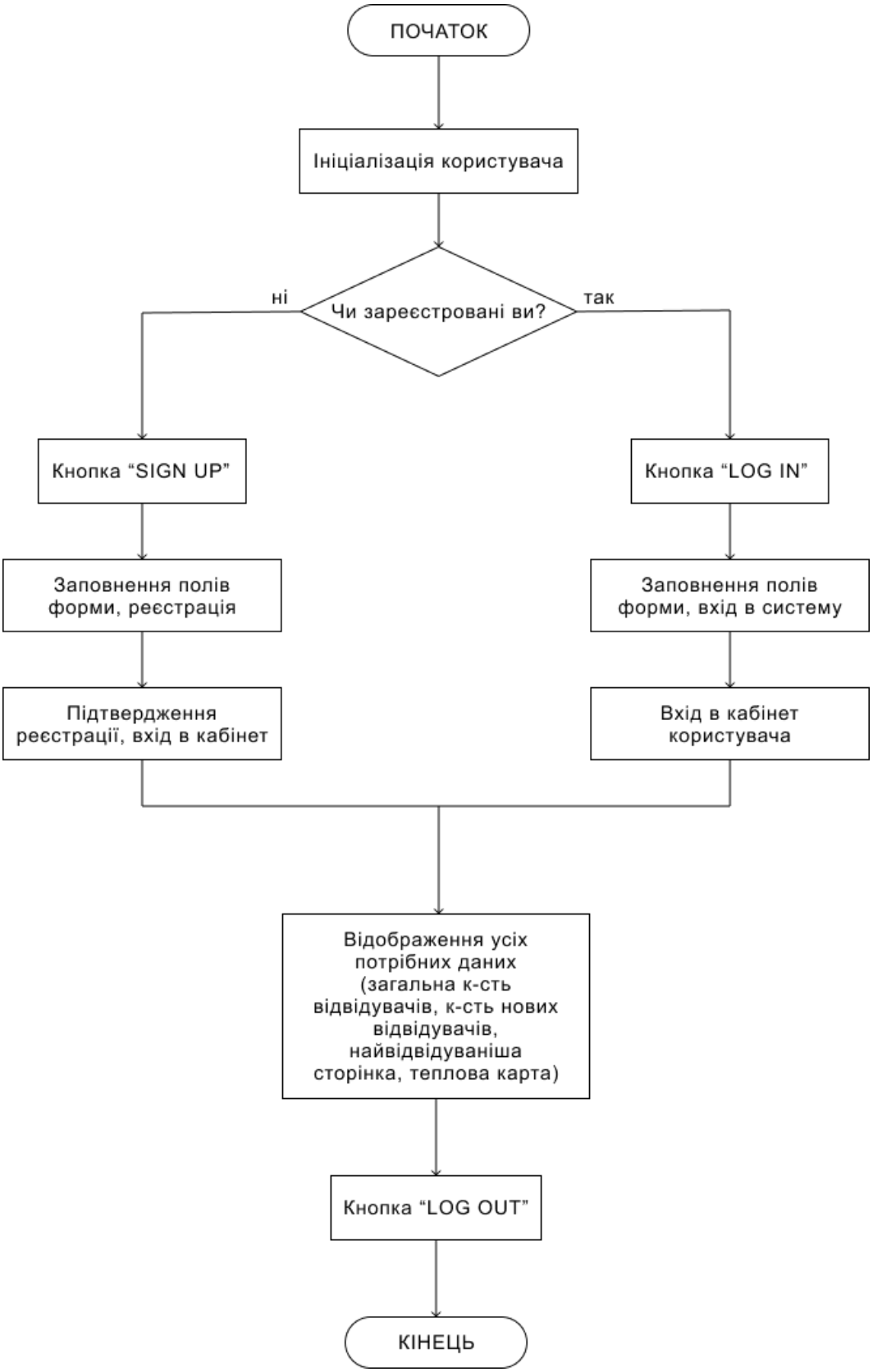

Рисунок 3.1 – Блок-схема алгоритму роботи системи.

У сучасних проєктах часто використовується мова UML (Unified Modelling Language) як інструмент для детального опису проєкту. UML включає в себе багато різноманітних діаграм, але найбільш цікавою для нас є діаграма прецедентів (Use Case).

Діаграма прецедентів є простою і містить невелику кількість можливих варіантів використання. Однак, якщо на діаграмі зображено більше 20 фігур, вона вважається неправильною. Основна мета використання цієї діаграми полягає в коректному відображенні динамічного аспекту системи.

Для збору основних вимог до системи використовується діаграма варіантів використання, яка включає як внутрішні, так і зовнішні впливи. До цих вимог можна віднести і вимоги розробки. Тобто, коли аналізують систему для збору всіх її можливих функціональних можливостей, спочатку виділяють прецеденти і визначають всіх можливих акторів.

Щоб побудувати діаграму перцедентів, спершу треба визначити основних діючих акторів та можливі варіанти використання.

Отже, у якості головного героя розроблювальної системи дослідження інформаційних систем виділимо користувача – це актор, який використовує систему для дослідження динаміки його сайту: перегляд кількості відвідувань, кількості користувачів, нових користувачів, виділення найбільш відвідуваних сторінок та теплова карта [10].

Другим кроком визначимо основні прецеденти проекту:

– можливість авторизуватися;
– реєстрація нового користувача;
– перегляд та аналіз отриманих даних динаміки сайту користувача.

На рис. 3.2 зображено діаграму прецедентів, яка представляє собою усі зв’язки акторів з можливими прецедентами.

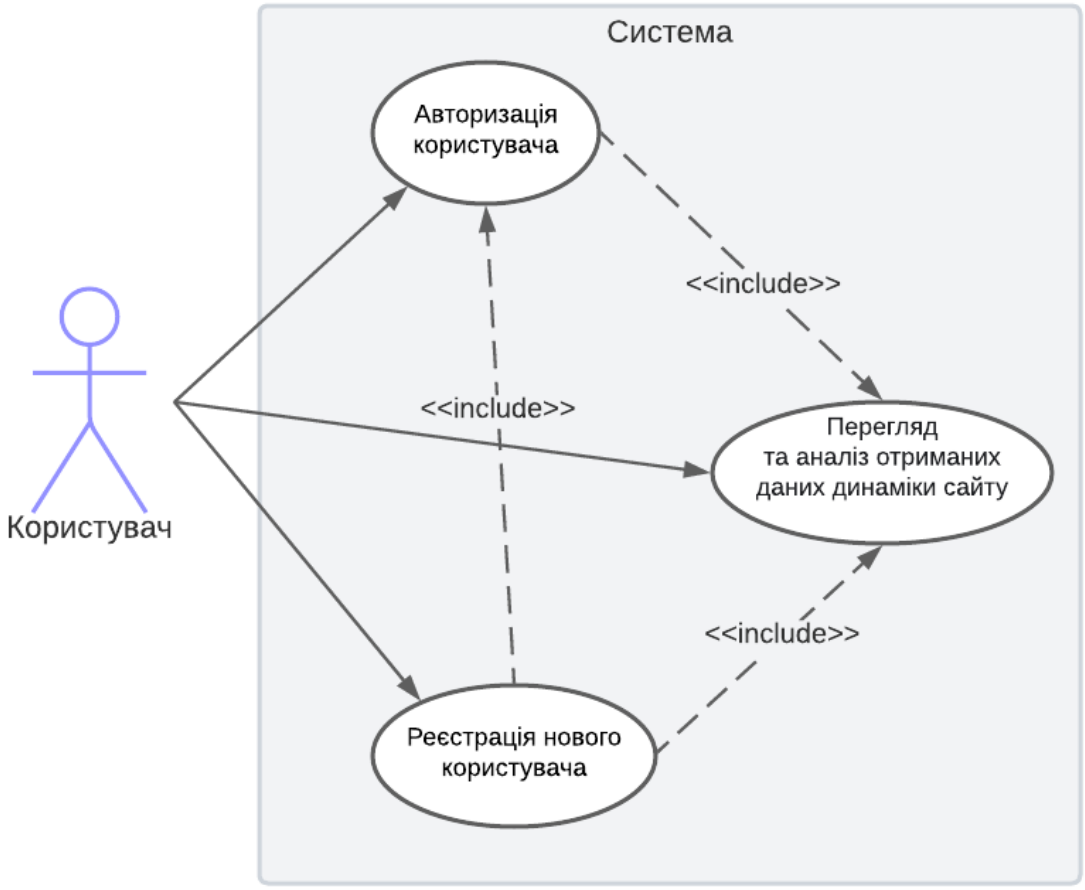

Рисунок 3.2 – UML діаграма прецедентів системи дослідження.

Ця діаграма дозволяє комунікувати між розробниками, замовниками та іншими учасниками проєкту, надаючи візуальне уявлення про функціональність системи. Вона допомагає визначити основні функції системи, ідентифікувати ролі користувачів та встановити зв'язки між ними [11].

3.2 Програми для розробки системи Figma

Перша програма яка допоможе нам при розробці нашої системи — це Figma (див. рис. 3.3). Figma є інструментом для дизайну і прототипування, що дозволяє командам працювати разом над проектами у реальному часі. Він набув популярності серед дизайнерів, розробників і продуктових менеджерів завдяки своїм можливостям та зручності використання.

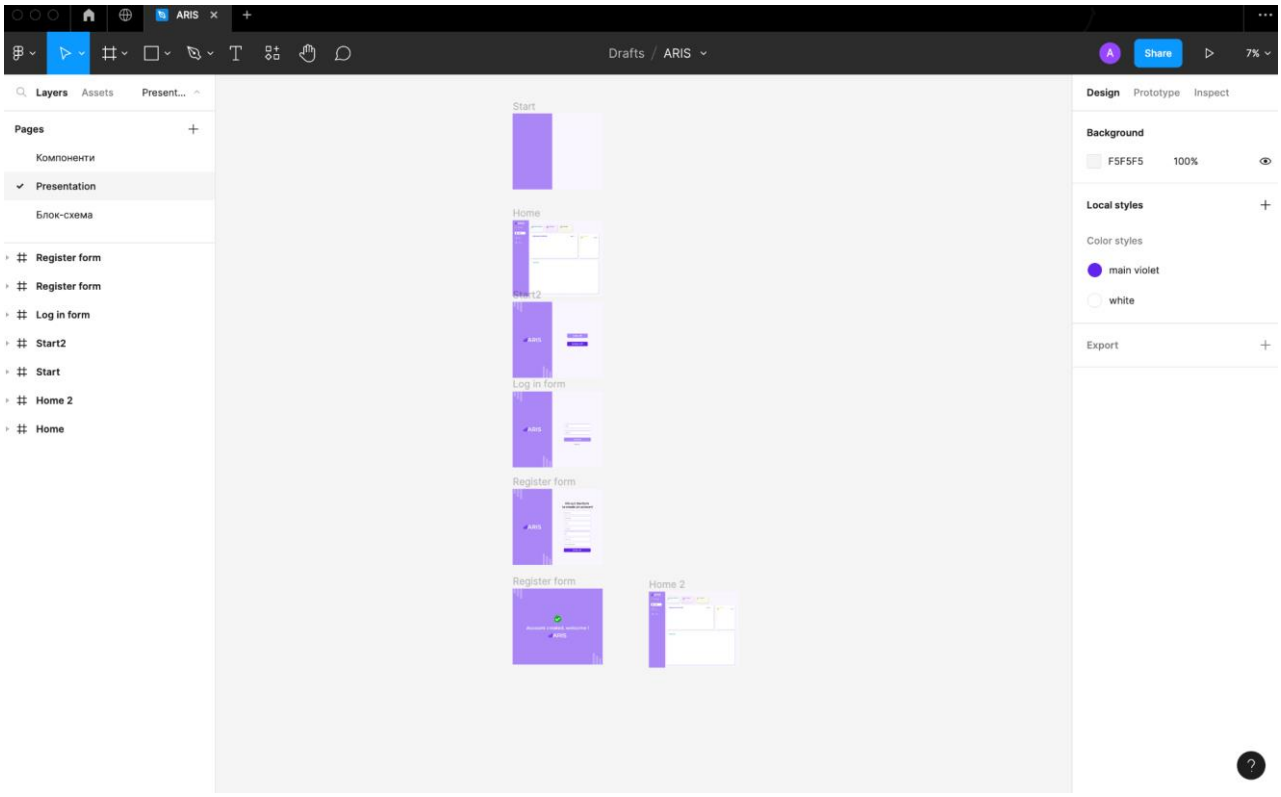

Рисунок 3.3 – Робочий інтерфейс програми Figma

Figma дозволяє створювати високоякісні макети, інтерактивні прототипи та спільно працювати над ними в режимі реального часу. Одна з головних переваг Figma полягає в тому, що він базується на хмарній технології, що дозволяє користувачам спільно редагувати дизайни без необхідності завантажувати або синхронізувати файли [12]. Тому ця програма дуже вдало підійде для реалізації прототипу нашої розроблювальної системи. Такий прототип допоможе в швидкій та якісній реалізації системи.

Visual Studio Code – це безкоштовний редактор коду, розроблений компанією Microsoft. Це один з найпопулярніших інструментів для редагування і написання коду серед розробників усього світу. VS Code пропонує розширену функціональність, високу продуктивність та велику кількість додаткових плагінів для розширення можливостей редактора (див. рис. 3.4).

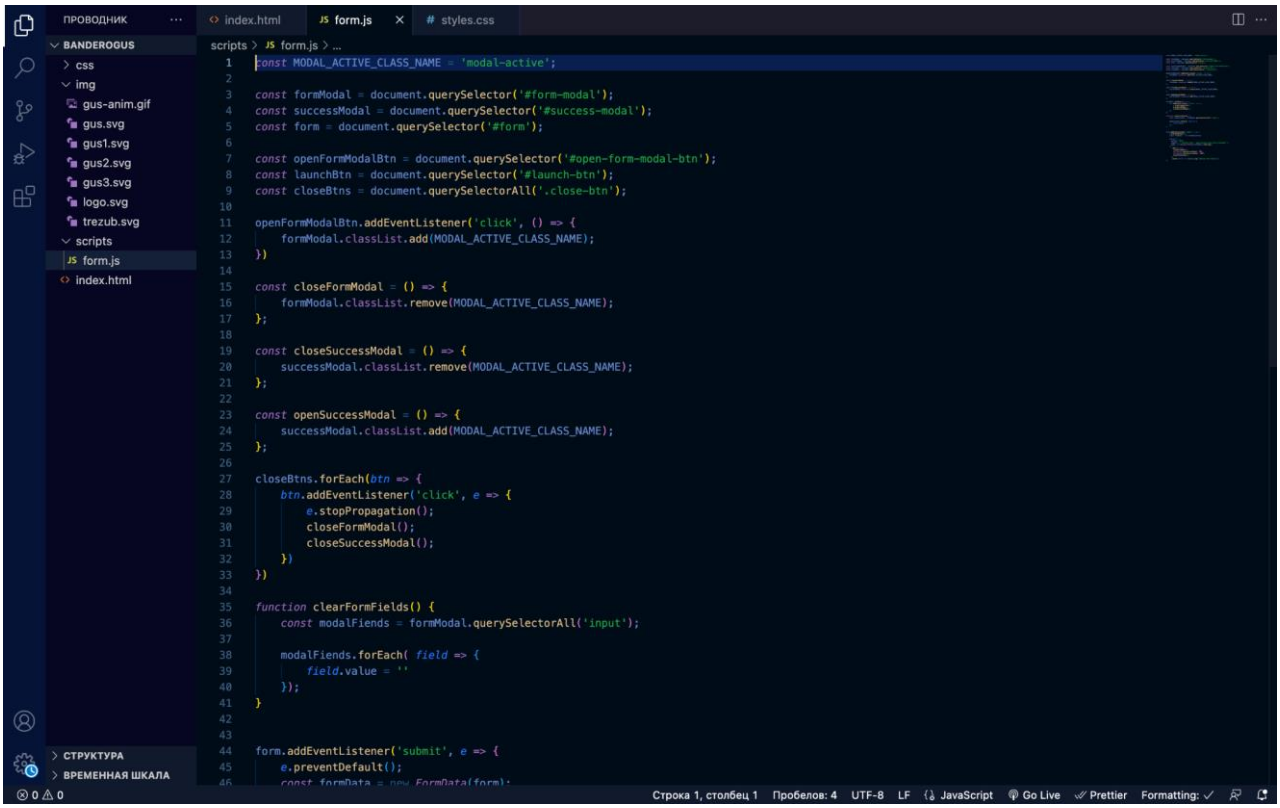

Рисунок 3.4 – Робочий інтерфейс програми Visual Studio Code

По-перше VS Code універсальний інструмент для розробки, тому що він доступний для всіх операційних систем. По-друге редактор підтримує велику кількість мов програмування і надає кольорове виділення синтаксису, що допомагає краще читати код. Крім того, ця програма пропонує автоматичне доповнення коду, що спрощує процес розробки та допомагає уникнути помилок. Вона підтримує інтеграцію з популярними системами контролю версій, такими як Git, що дозволяє зручно працювати з репозиторіями.

Програма має досить широкий вибір розширень, що дозволяють налаштовувати редактор під свої потреби. Редактор підтримує можливість відлагодження коду для різних мов програмування з використанням точок зупину та інших інструментів відлагодження. VS Code має вбудований термінал, що дозволяє взаємодіяти з командним рядком безпосередньо з редактора [13].

Visual Studio Code є потужним та дуже популярним інструментом для розробки програмного забезпечення, який пропонує багато корисних функцій та

можливостей для зручної та продуктивної роботи з кодом. Тому він ідеально підійде для вирішення завдань будь-якої складності.

phpMyAdmin – це безкоштовний веб-інтерфейс, призначений для управління та адміністрування баз даних MySQL. Він надає зручну та інтуїтивно зрозумілу середу для виконання різноманітних завдань, пов'язаних з базами даних, таких як створення, зміна, видалення таблиць, виконання SQL-запитів, імпорт та експорт даних, керування користувачами і їхніми привілеями, оптимізація та налагодження запитів, і багато іншого (див.рис.3.5).

phpMyAdmin зручний в управлінні базами даних, тому що працює через веб-браузер з будь-якого комп'ютера з доступом до Інтернету. Він надає широкий спектр функцій для управління базами даних, таких як створення, редагування та видалення таблиць, індексів, ключів, перегляд та редагування даних, виконання складних SQL-запитів, імпорт та експорт даних у різних форматах, тощо. Інтерфейс phpMyAdmin добре організований і інтуїтивно зрозумілий. Він надає швидкий доступ до різних функцій та опцій, що спрощує роботу з базами даних.

phpMyAdmin надає можливості для керування безпекою баз даних, включаючи налаштування користувачів, надання прав доступу, шифрування підключення до бази даних та інші заходи безпеки. Безпека даних — одна з основних вимог при створенні будь-якої системи.

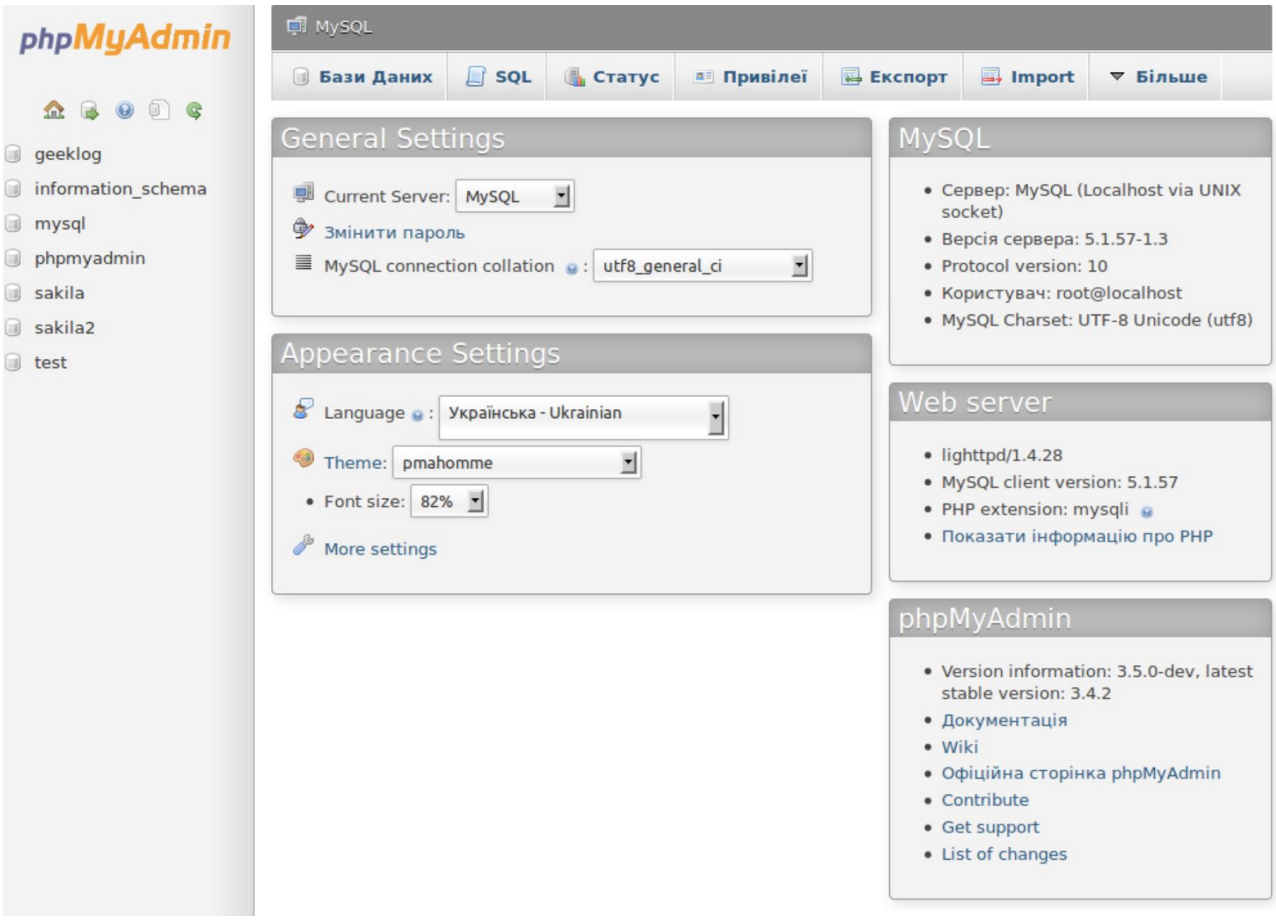

Рисунок 3.5 – Робочий інтерфейс програми phpMyAdmin

phpMyAdmin - це потужний інструмент для роботи з базами даних MySQL, який дозволяє легко та зручно керувати вашими базами даних за допомогою веб-інтерфейсу [14]. Він широко використовується розробниками, адміністраторами баз даних та веб-хостинг-провайдерами для управління та підтримки баз даних MySQL. Тому ця програма ідеально підійде для створення та управління базами даних.

3.3 Прототипування та розробка системи дослідження інформаційних систем

Прототипування (англ. prototyping) — це процес створення прототипу або моделі продукту для оцінки його функціональності, дизайну та взаємодії з користувачем до фінального виробництва. Прототипування є важливою частиною

розробки продукту, незалежно від його типу, будь то програмне забезпечення, апаратний засіб або фізичний предмет.

Головною метою прототипування є тестування та збирання зворотного зв'язку щодо продукту з реальними користувачами або зацікавленими сторонами. Це дозволяє виявити проблеми, недоліки та можливості для вдосконалення ще на ранніх етапах розробки. Прототипи можуть бути створені у різних форматах, включаючи провідні моделі, макети, інтерактивні демонстрації або функціональні прототипи з мінімальним набором функцій.

Прототипування дозволяє команді розробників отримати реальний зв'язок із користувачами та отримати важливі відгуки для поліпшення продукту перед його фінальним випуском. Воно також сприяє уточненню вимог, визначенню функціональності та інтерфейсу, а також може служити для оцінки та залучення інвесторів або партнерів. Прототипування є важливою складовою процесу розробки, яка допомагає зменшити ризики та забезпечити успішну реалізацію продукту.

Для прототипування нашої розроблювальної системи дослідження інформаційних систем було обрано програму Figma. Назва нашої системи ARIS, що є скороченням від повної назви роботи — *Automated research of information systems*. Перед початком прототипування важливо повторно визначити функціонал нашої системи, щоб не пропустити жодних деталей, та правильно розробити прототип, для подальшого легкої програмної реалізації системи. І так, основний функціонал який матиме наша розроблювальна система:

– авторизація користувача (можливість реєстрації нового);

– збір даних про відвідування (кількість відвідувачів, відвідування сторінок);

– аналіз поведінки користувачів (найвідвідуваніша сторінка);

– теплові карти (область сайту на яку користувачі найбільше звертають увагу);

– візуалізація отриманих даних.

Розпочнемо прототипування з першого кроку, авторизація або реєстрація користувача. Користувач повинен мати два варіанти входження у систему, перший варіант це коли він не зареєстрований — це «SIGN UP», а другий варіант коли користувач вже зареєстрований — це «LOG IN» (див.рис.3.6).

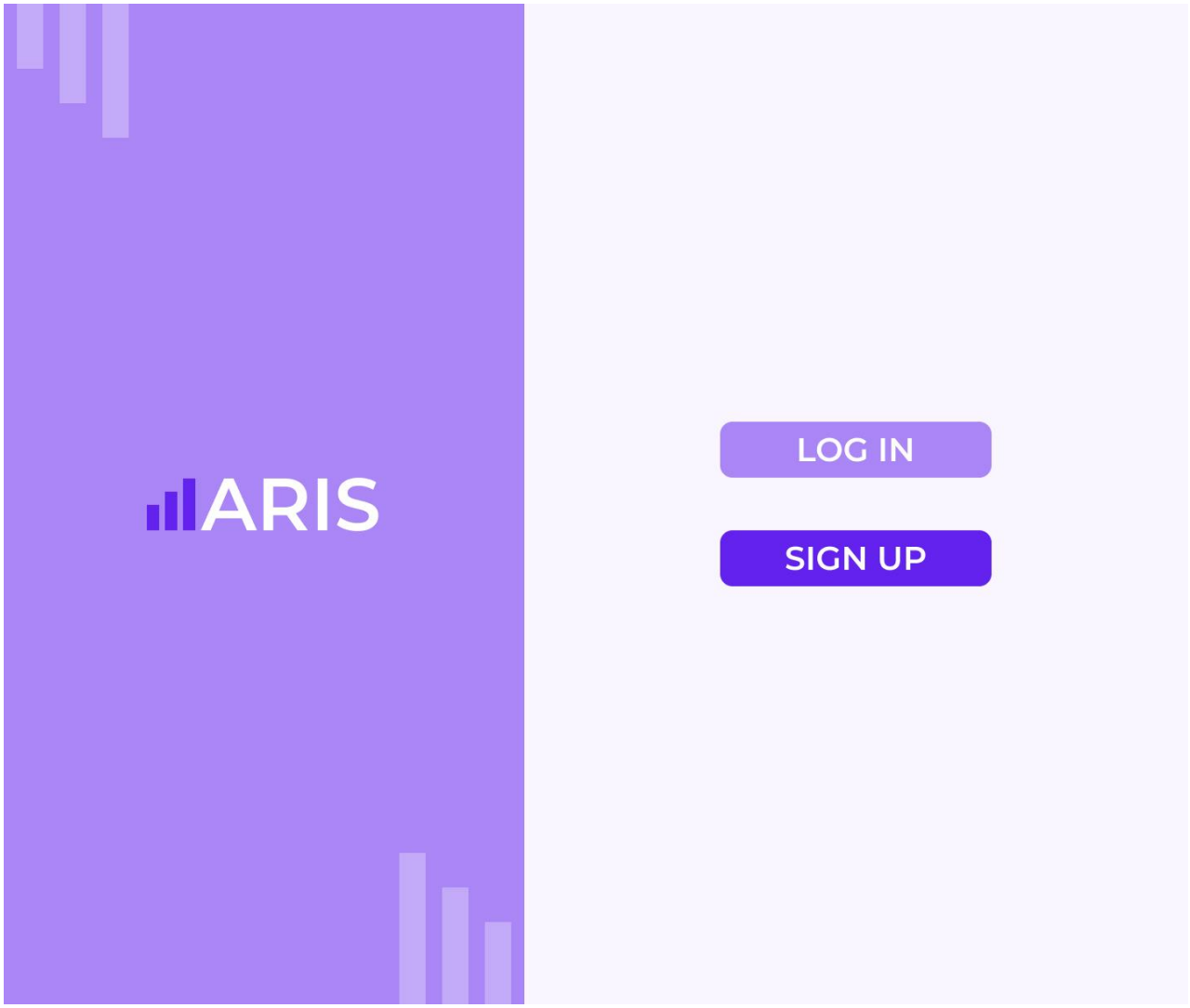

Рисунок 3.6 – Головний екран розроблювальної системи ARIS

Як тільки користувач обирає зручний для себе варіант, він переходить до наступного кроку. Розглянемо перший варіант – «SIGN UP», коли користувач не зареєстрований. Для реєстрації користувача необхідно заповнити певний перелік даних. По-перше, це дані про ім'я та прізвище користувача, які ідентифікують його особу. По-друге, адреса електронної пошти, яка використовується для зв'язку з

користувачем та надсилання повідомлень. Назва компанії, можливо це буде назва інтернет магазину, або у користувача свій сайт-блог та ін. Один з найголовніших пунктів це посилання на сайт, саме з цього посилання наша система буде брати усі необхідні дані для їх відображення. Ну і звичайно пароль для входження в систему (див. рис. 3.7)

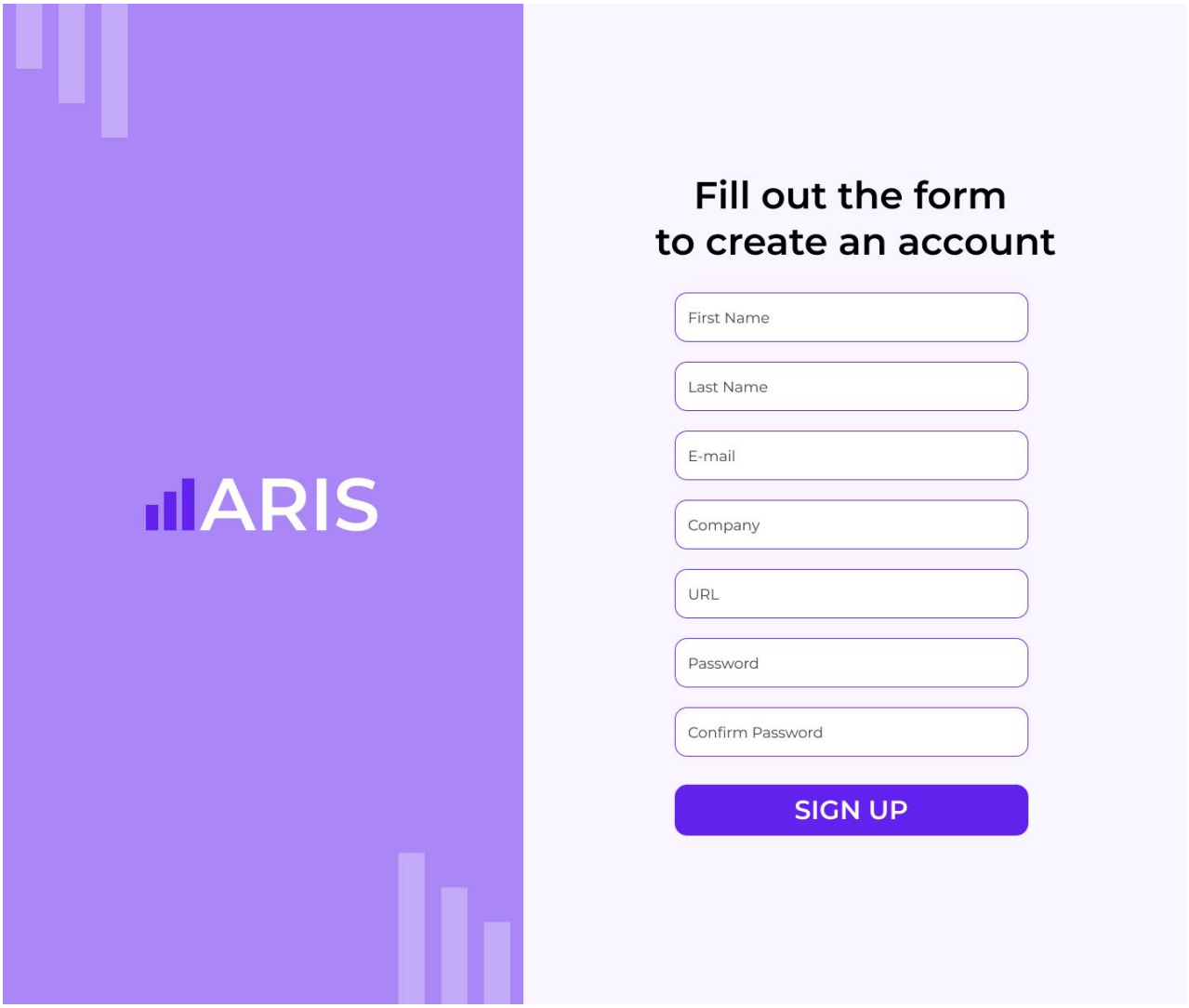

Рисунок 3.7 – Екран реєстрації в системі ARIS

Після реєстрації варто повідомити користувача про успішну або невдалу реєстрацію у нашій системі (див. рис. 3.8).

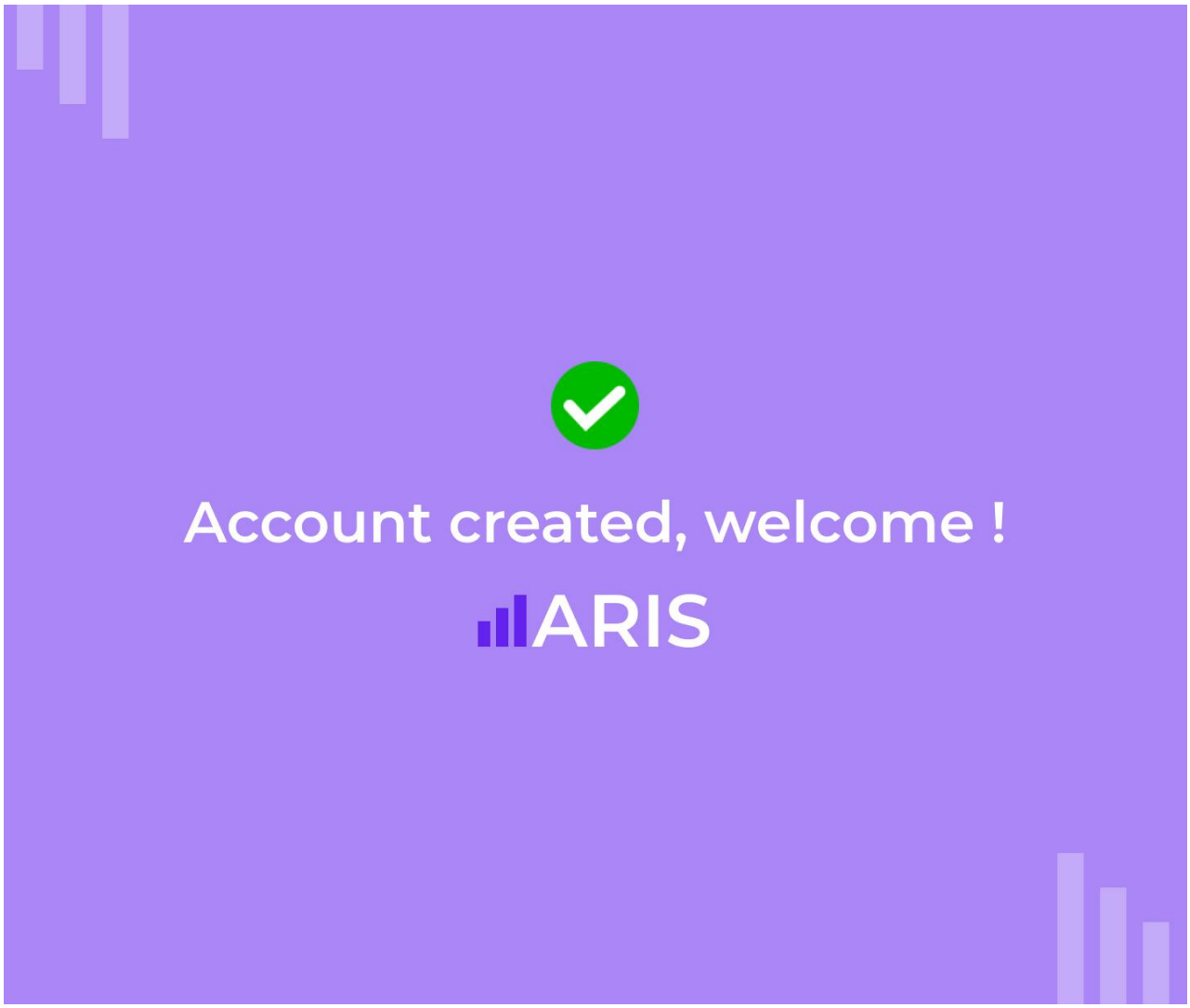

Рисунок 3.8 – Екран успішної реєстрації в системі ARIS

Тепер розглянемо другий варіант — «LOG IN» (Додаток Г), коли користувач зареєстрований. У цьому випадку форма входу досить проста, це електронна адреса та пароль входу (див.рис.3.9). Але крім того, варто врахувати варіант, що людина не зареєстрована і випадково натиснула кнопку «LOG IN». У цьому випадку ми додамо невелику кнопку в кінці форми «Register?», для того щоб користувач міг натиснути і перейти на екран реєстрації користувача рис.3.8.

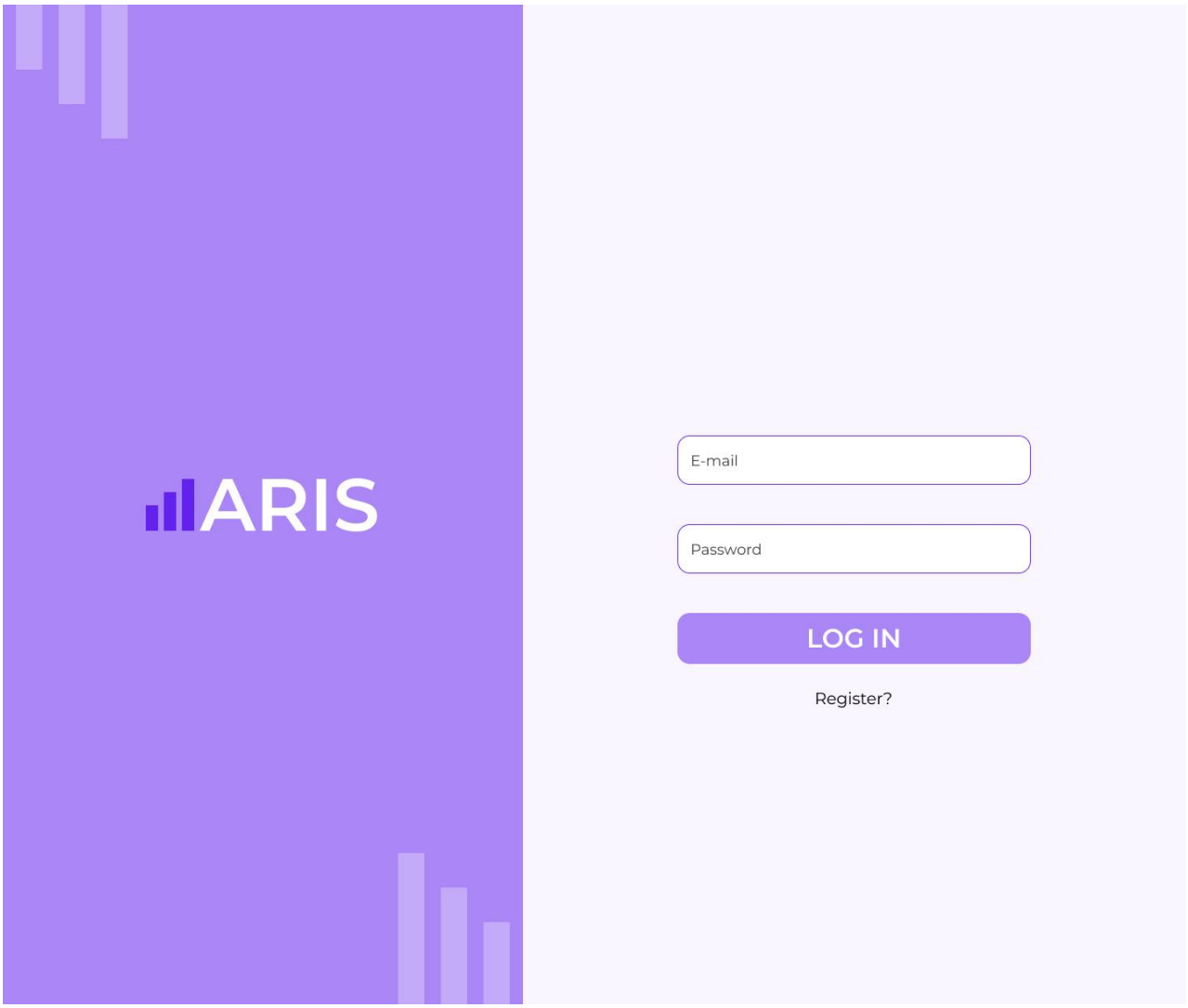

Рисунок 3.9 – Екран авторизації зареєстрованого користувача в системі ARIS

Після успішної реєстрації або авторизації, користувачу відкривається головне вікно системи, де користувач бачить всю інформацію про роботу свого сайту та відображає увесь перелік функціоналу розроблювальної системи. А саме: загальна кількість відвідувань (у відповідний день чи місяць), загальна кількість користувачів, кількість нових користувачів, перелік найбільш відвідуваних сторінок від найбільш відвідуваної до найменш і теплова карта сайту. Крім того в кабінеті передбачено майбутню сторінку FAQs, де користувач зможе знайти для себе інформацію на яку шукає питання, ну і звичайно кнопка виходу «LOG OUT», що перекидає користувача на початковий екран (див.рис.3.10).

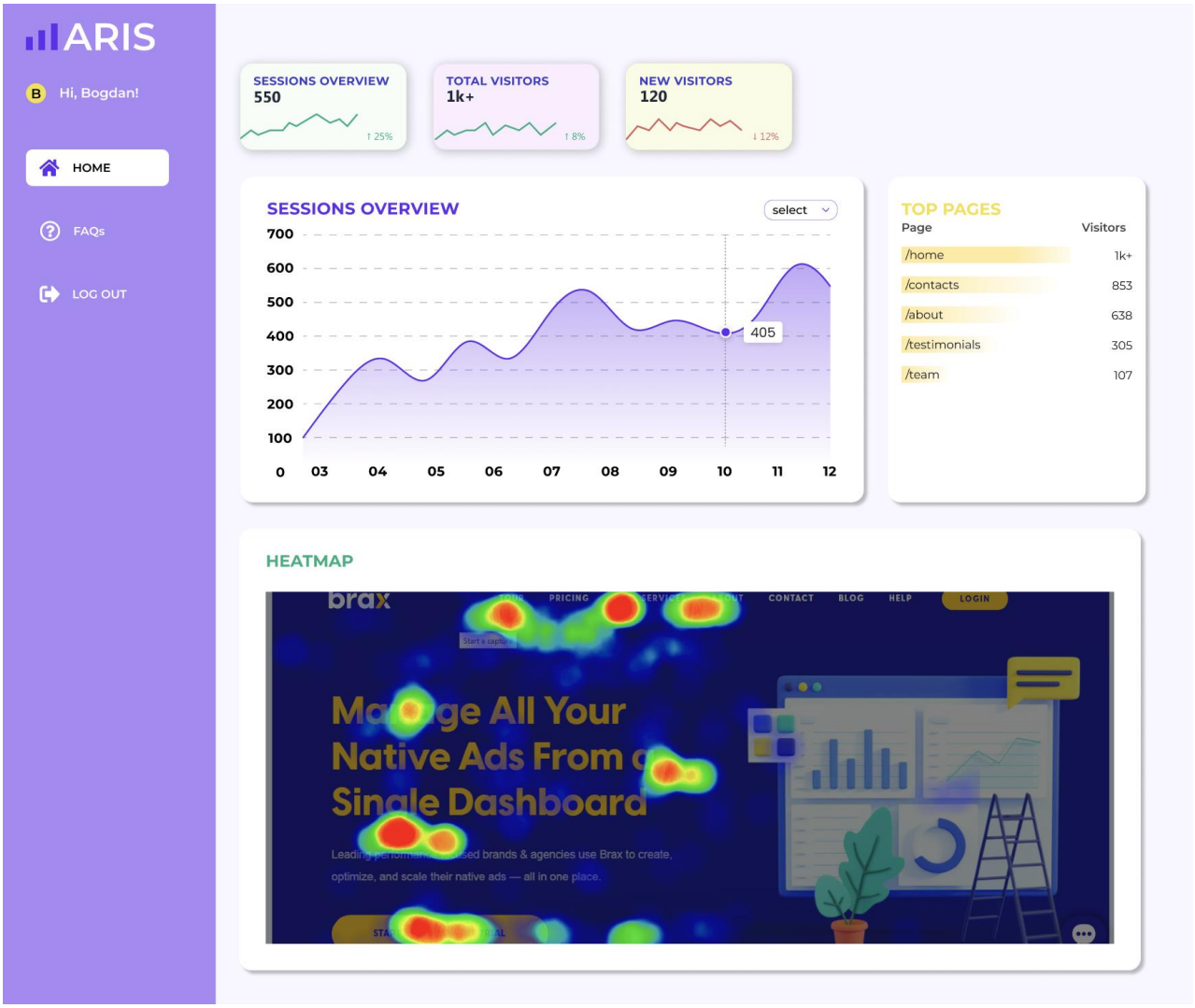

Рисунок 3.10 – Кабінет користувача в системі ARIS

Тепер розглянемо реалізацію нашого головного функціоналу розроблювальної системи через програму Visual Studio Code. Перша функція це збір даних про відвідування (кількість відвідувачів, відвідування сторінок), цю функцію реалізуємо через Алгоритм Cookie (Додаток В), він досить поширений і ефективний для визначення кількості відвідувачів сайту. У лівій частині кабінету на графіку відвідувань передбачено фільтр, де можна обрати вибірку за місяць або за останні 10 днів (див. рис. 3.11).

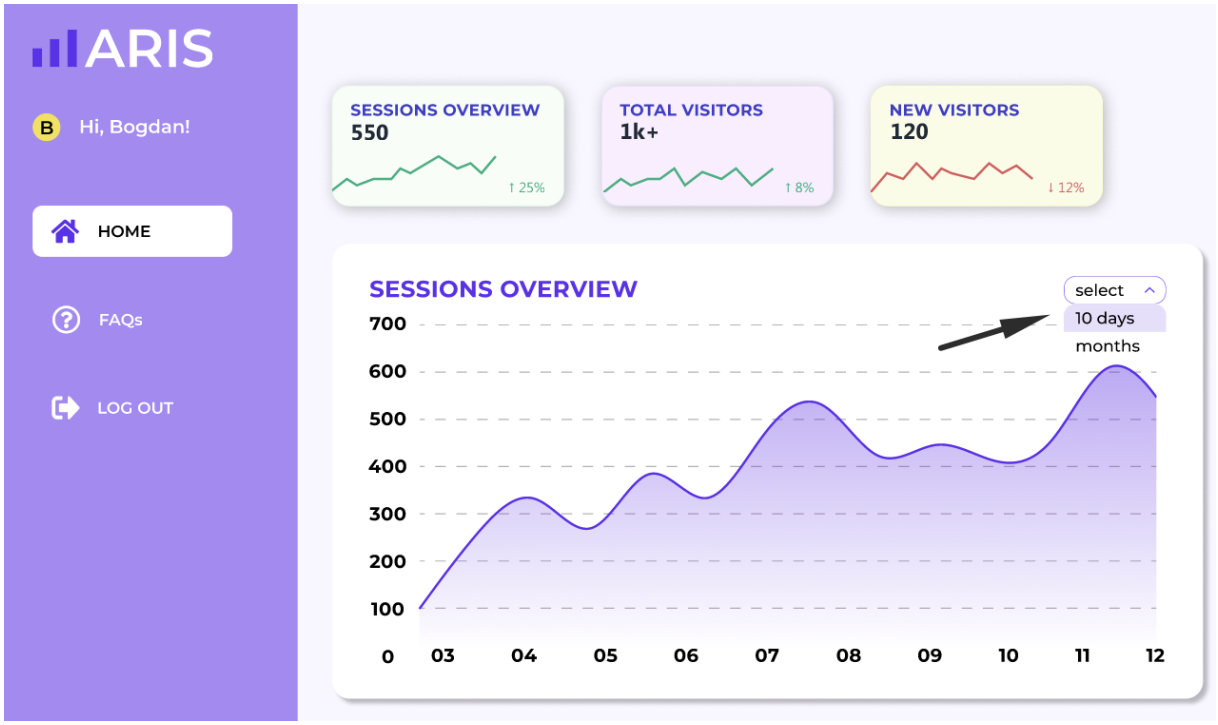

Рисунок 3.11 – Фільтр для графіку відвідувань

Також при наведенні на відповідну дату/місяць, можна побачити скільки відвідувачів було на нашому сайту цього дня/місяця (див.рис.3.12).

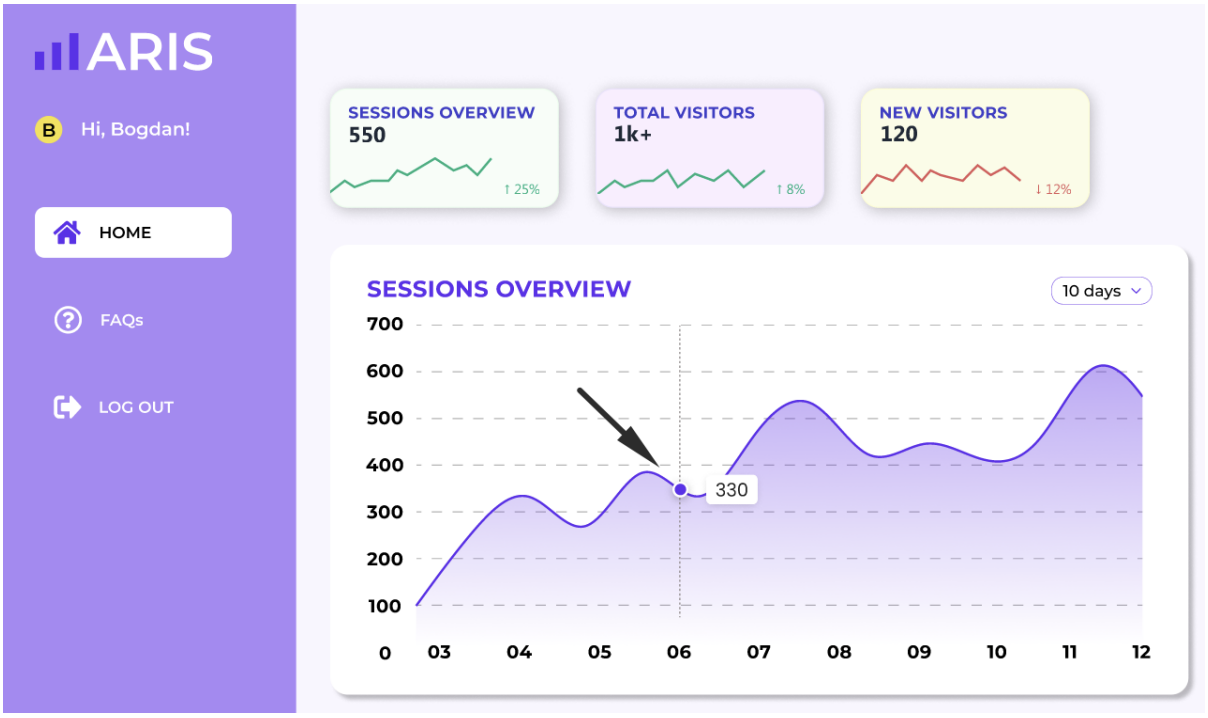

Рисунок 3.12 – Відображення кількості відвідувачів сайту.

Тепер розглянемо реалізацію другої функції аналіз поведінки користувачів (найвідвідуваніша сторінка). Цю функцію реалізуємо через Алгоритм PageRank (Додаток А). У правій частині кабінету зображено список сторінок сайту, які розташовані шляхом спадання від найбільш відвідуваної до найменш, крім того зазначена кількість користувачів що відвідували сайт, а кольором виділений прогрес відвідування, на основі кількості відвідувачів (див.рис.3.13).

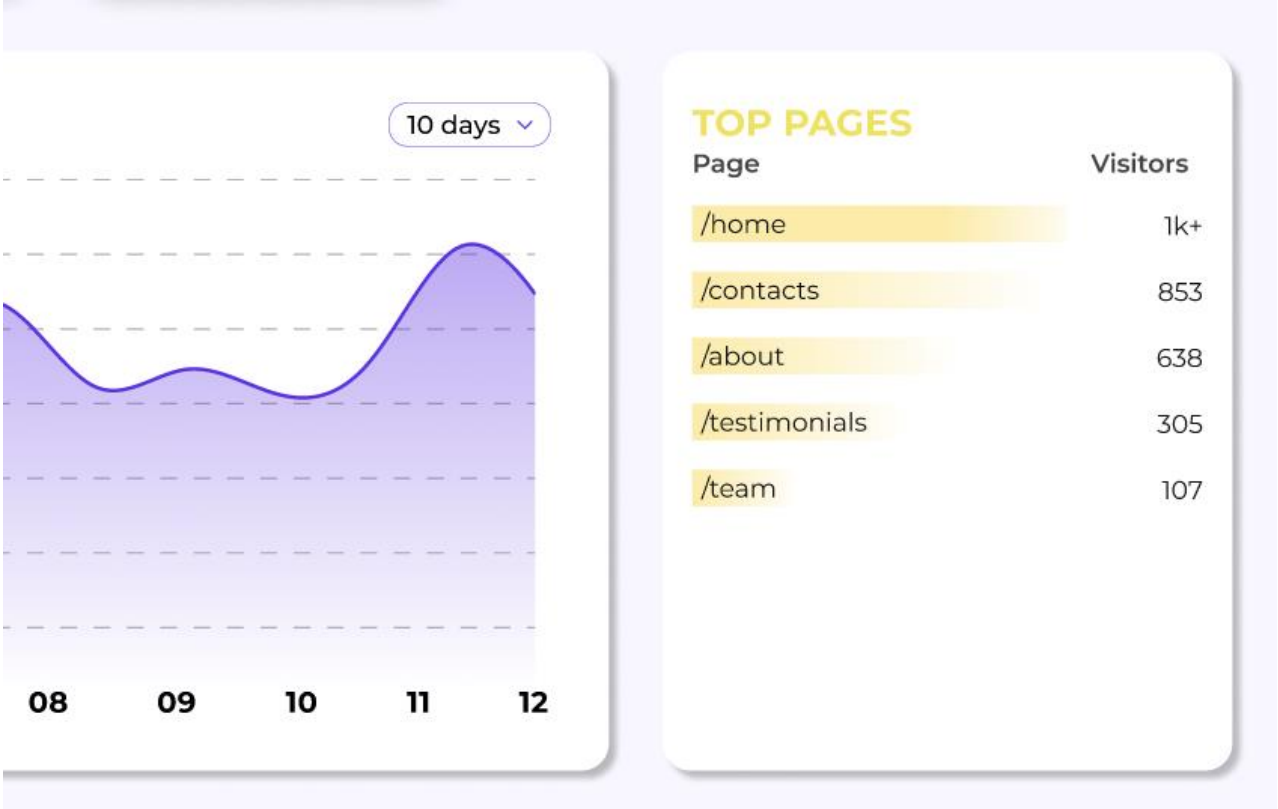

Рисунок 3.13 – Відображення рейтингу сторінок по їх відвідуванню.

І остання функція нашої розроблювальної системи це теплова карта (область сайту на яку користувачі найбільше звертають увагу). Цю функцію реалізуємо через JavaScript-бібліотеку Heatmap.js (Додаток Б). Теплові карти дзволяють швидко виявити гарячі точки, зони низького або високого значення, градієнти або аномалії. Таким чином, вони допомагають виявити невидимі залежності та зробити висновки на основі візуальних зображень (див.рис.3.14).

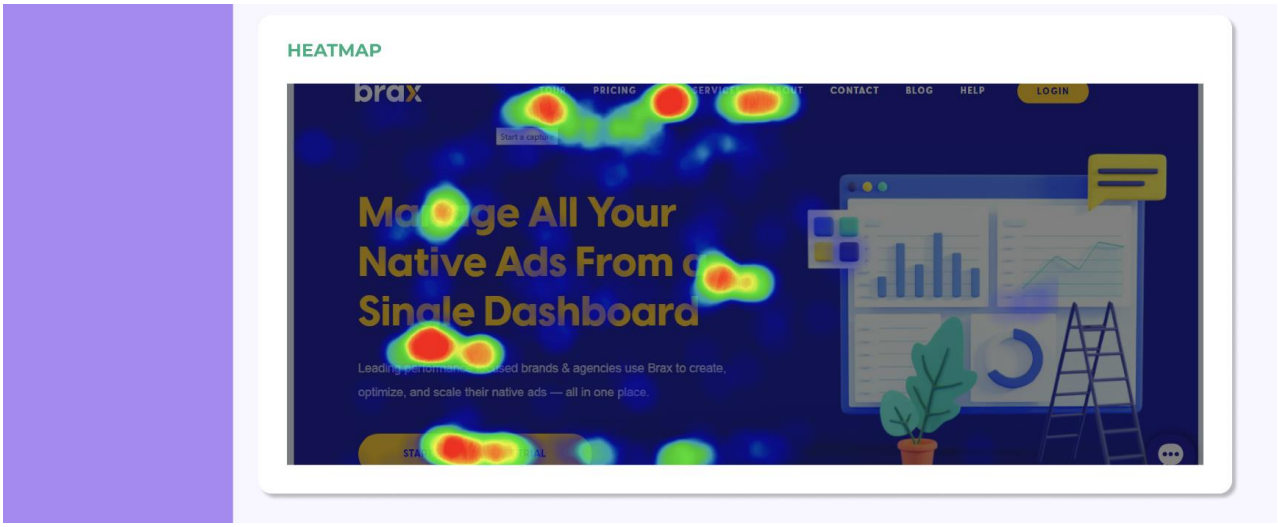

Рисунок 3.14 – Теплова карта.

Меню в кабінеті користувача досить просте і складається всього з трьох пунктів: HOME — домашня сторінка, FAQs — сторінка часто поставлених питань та LOG OUT — повернення на першу сторінку системи (див.рис.3.15).

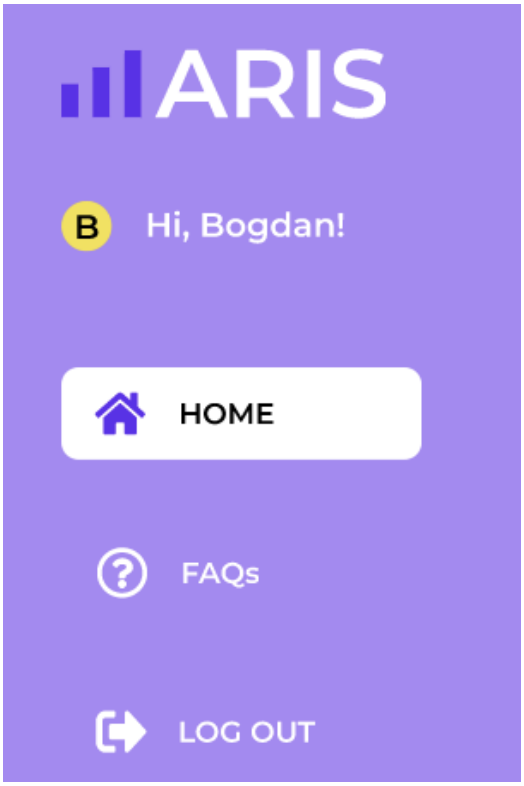

Рисунок 3.15 – Меню кабінету користувача.

Висновки до 3 розділу

Отже, в третьому розділі визначено основні функціональні можливості автоматизованої системи дослідження інформаційних систем: авторизація користувача (можливість реєстрації нового), збір даних про відвідування, аналіз поведінки користувачів та теплові карти.

Блок-схемою схематично зображено детальний процес роботи автоматизованої системи. Для більш точної та зрозумілої роботи нашої системи було створено UML діаграму прецедентів, де відображена взаємодія користувача з прецедентами автоматизованої системи.

Також, розглянули та визначили основні програми, котрі будуть використані при розробці системи. Figma — для дизайну та прототипування системи, Visual Studio Code — для розробки системи, phpMyAdmin — для роботи з базами даних.

Крім того, зобразити та показали покрокову роботу системи через прототип та реалізували окремі головні функції автоматизованої системи дослідження інформаційних систем.

ВИСНОВОК

У результаті виконання кваліфікаційної роботи розроблено прототип автоматизованої системи дослідження інформаційних систем та реалізовано окремі головні функції системи. Проведено огляд предметної сфери, визначено основне поняття ІС, основні процеси, що лежать в роботі інформаційної системи. Виконано огляд систем, що мають схожі функції та вирішують такі ж задачі, що поставленні перед нашою системою. Визначено та вирішено основні задачі системи: авторизація користувача (можливість реєстрації нового), збір даних про відвідування, аналіз поведінки користувачів та теплові карти.

Розглянуто різні методи та технології для реалізації функціоналу системи дослідження інформаційних систем. Запропоновано декілька методів та алгоритмів для реалізації основного функціоналу серед яких виділили основі, котрі були використані при розробці нашої системи.

Було створено блок-схему для схематичного зображення процесу роботи автоматизованої системи та UML діаграму прецедентів, де відображена взаємодія користувача з прецедентами автоматизованої системи.

У ході тестування прототипу було перевірено відповідність системи функціональним вимогам. Тестування було успішно виконано. Дана система повністю відповідає всім відповідним вимогам. Також, було додано код реалізації функцій системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Структура інформаціної системи. URL: http://ck.vk.mnau.edu.ua/ck/portfolio_Macovey/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_IKT/%D0%A1%D0%B0%D0%BC_%D1%80%D0%BE%D0%B1/%D1%81%D0%B0%D0%BC_%D1%80%D0%BE%D0%B11.htm (дата звернення 04.04.2023).

2. Joseph Valacich, Christoph Schneider. Information Systems Today: Managing the Digital World. Pearson, 2017.

3. Components of an Information System. URL: https://www.mbaknol.com/management-information-systems/components-of-an-information-system/ (дата звернення 07.04.2023).

4. What is Information System? URL: https://emeritus.org/in/learn/information-system/ (дата звернення 12.04.2023).

5. Page Rank Algorithm. URL: https://towardsdatascience.com/pagerank-algorithm-fully-explained-dc794184b4af (дата звернення 18.04.2023).

6. What is a Heatmap? URL: https://www.fullstory.com/heatmap/ (дата звернення 26.04.2023).

7. Heatmap.js Documentation. URL: https://www.patrick-wied.at/static/heatmapjs/docs.html (дата звернення 28.04.2023).

8. Шнайдер Р. Microsoft SQL Server 6.5. Проектування високопродуктивних баз даних, – К.:, 2010. – 361 с.

9. What is MySQL? URL: https://www.oracle.com/mysql/what-is-mysql/#:~:text=MySQL%20Database%20is%20a%20client,%2Dprogramming%20interfaces%20(APIs). (дата звернення 12.05.2023).

10. Unified Modeling Language (UML) | An Introduction URL: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/ (дата звернення: 29.05.2023).

11. Chonoles M. J., Schardt J.A. UML 2 for Dummies. – Hungry Minds, 2003. – 412 p. 7. (дата звернення: 29.05.2023).

12. Що таке Figma: функції, інструменти та переваги. Євген Омельчук URL: https://wezom.academy/ua/chto-takoe-figma-funktsii-instrumenty-ipreimuschestva/ (дата звернення 05.06.2023).

13. Скляр Д. Вивчаємо PHP 7: керівництво по створенню інтерактивних веб-сайтів.: Пер. з англ. - СпБ. : ООО "Альфа-книга", 2017. - 464 с. (дата звернення 08.06.2023).

14. Marc Delisle, Mastering Phpmyadmin 3.3.X for Effective MySQL Management – , 2014. – 412 p. (дата звернення 12.06.2023).

15. Фрімен Е., Фрімен Е. Вивчаємо HTML, XHTML та CSS.: пер. з англ.б, 2015 (дата звернення 15.06.2023)

16. Paul DuBois, MySQL Cookbook, 3rd Edition – O'Reilly Media, Inc., 2014. – 412 p. (дата звернення 15.06.2023)

17. Peter Morville, Information Architecture for the World Wide Web: Designing Large-Scale Web Sites – O'Reilly Media, Inc., 2020. – 380 p. (дата звернення 15.06.2023)

18. Ramez Elmasri, Shamkant B. Navathe, Fundamentals of Database Systems, 2018 (дата звернення 16.06.2023)

19.

ДОДАТОК А

Алгоритм PageRank

```
//Expose our library to be called externally
module.exports = function (nodeMatrix, linkProb, tolerance, callback, debug) {
   if (!nodeMatrix || !linkProb || !tolerance || !callback)
      throw new Error("Provide 4 arguments: "+
         "nodeMatrix, link probability, tolerance, callback")
         if (!debug) debug=false
   return new Pagerank(nodeMatrix, linkProb, tolerance, callback, debug)
}
// INITIALIZE
function Pagerank (nodeMatrix, linkProb, tolerance, callback, debug) {
   this.outgoingNodes = nodeMatrix
   this.linkProb = linkProb
   this.tolerance = tolerance
   this.callback = callback
   this.pageCount = this.outgoingNodes.length
   this.coeff = (1-linkProb)/this.pageCount

   this.probabilityNodes = []
   this.incomingNodes = []
   this.debug=debug

   this.startRanking()
}
//START RANKING
Pagerank.prototype.startRanking = function () {
   var initialProbabilty = 1/this.pageCount
      , outgoingNodes = this.outgoingNodes
      for (i in outgoingNodes) {
      this.probabilityNodes.push(initialProbabilty)
      for (a in outgoingNodes[i]) {
         var index = outgoingNodes[i][a]
         if (!this.incomingNodes[index]) this.incomingNodes[index]=[]
         this.incomingNodes[index].push(i)
      }
   }
   if (this.debug) this.reportDebug(1)
```

```
    this.iterate(1)
}
//LOG ITERATION TO CONSOLE
Pagerank.prototype.reportDebug = function (count) {
    console.log("Pages: " + this.outgoingNodes.length)
    console.log(this.incomingNodes)
    console.log("____ITERATION "+count+"____")
    console.log(this.probabilityNodes)
}
//CALCULATE NEW WEIGHTS
Pagerank.prototype.iterate = function (count) {

    var result = []
    for (b in this.probabilityNodes) {
        var sum = 0
        for (var a=0; a<this.incomingNodes[b].length; a++) {
            prob = this.probabilityNodes[this.incomingNodes[b][a]]
            ct = this.outgoingNodes[this.incomingNodes[b][a]].length
            sum += (prob/ct)
        }
         var res = this.coeff+this.linkProb*sum
            , max = this.probabilityNodes[b]+this.tolerance
            , min = this.probabilityNodes[b]-this.tolerance
            if (min <= res >= max) result.push(res)
             this.probabilityNodes[b]=res
    }
     if (result.length == this.pageCount)
        return this.callback(null, result)
        if (this.debug) this.reportDebug(count)

    this.iterate(++count)
}
```

ДОДАТОК Б

Загальна структура коду JavaScript-бібліотека Heatmap.js

```
//Підключення бібліотеки
<script src="path/to/heatmap.js"></script>
//Створення екземпляру теплової карти:
var heatmapInstance = h337.create({
  container: document.getElementById('heatmapContainer'), // елемент, в якому буде відображатися карта
  // інші налаштування
});
//Встановлення даних для карти
var data = {
  max: 10, // максимальне значення для шкали кольорів
  data: [
    { x: 100, y: 200, value: 5 }, // точки з координатами та значеннями
    { x: 150, y: 250, value: 8 },
    // додаткові точки
  ]
};
heatmapInstance.setData(data);
//Налаштування вигляду карти:
heatmapInstance.setOptions({
  radius: 20, // радіус округлих областей вокруг кожної точки
  opacity: 0.6, // прозорість карти
  gradient: {
    '0.4': 'blue', // кольорова палітра
    '0.6': 'green',
    '0.8': 'yellow',
    '1.0': 'red'
  },
  // інші налаштування
});
```

ДОДАТОК В

Код алгоритму Cookies

```
// Отримання значення кукі
function getCookie(name) {
  const cookieValue = document.cookie.match(`(^|;)\\s*${name}\\s*=\\s*([^;]+)`);
  return cookieValue ? cookieValue.pop() : '';
}

// Встановлення кукі
function setCookie(name, value, days) {
  const expirationDate = new Date();
  expirationDate.setTime(expirationDate.getTime() + (days * 24 * 60 * 60 * 1000));
  const expires = `expires=${expirationDate.toUTCString()}`;
  document.cookie = `${name}=${value}; ${expires}; path=/`;
}

// Перевірка наявності кукі
function checkCookie() {
  const cookieName = 'visitor';
  const visitorId = getCookie(cookieName);

  if (visitorId !== '') {
    // Кукі вже існує, відвідувач не рахується як новий
    console.log('Visitor already exists');
  } else {
    // Кукі не існує, встановлюємо новий кукі
    const newVisitorId = generateUniqueId(); // Генеруємо унікальний ідентифікатор
    setCookie(cookieName, newVisitorId, 365); // Встановлюємо кукі на 365 днів
    console.log('New visitor created');
  }
}

// Генерування унікального ідентифікатора
function generateUniqueId() {
  // Реалізація генерації унікального ідентифікатора за потреби
  // Можна використовувати, наприклад, timestamp або рандомні значення
  return 'unique_id';
}
// Виклик функції перевірки кукі
checkCookie();
```

ДОДАТОК Г

Форма авторизації користувача

index.html

```
 <link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet' type='text/css'>

<div class="login">
 <h2 class="active"> LOG IN </h2>

 <h2 class="nonactive"> SIGN UP </h2>
 <form>

  <input type="text" class="text" name="username">
   <span>E-mail</span>
  <br>
  <br>

  <input type="password" class="text" name="password">
  <span>Password</span>
  <br>

  <button class="signin"> LOG IN </button>
  <hr>

  <a href="#">Register?</a>
 </form>

</div>
```

styles.css

```
body,
form {
 padding-top: 80px;
}

.active {
 border-bottom: 2px solid #1161ed;
}

.nonactive {
 color: rgba(255, 255, 255, 0.2);
}

h2 {
 padding-left: 12px;
 font-size: 22px;
 text-transform: uppercase;
 padding-bottom: 5px;
 letter-spacing: 2px;
 display: inline-block;
 font-weight: 100;
}

h2:first-child {
 padding-left: 0px;
}

span {
 text-transform: uppercase;
 font-size: 12px;
```

```
  opacity: 1;
  color: #111;
  display: inline-block;
  position: relative;
  top: -70px;
  transition: all 0.5s ease-in-out;
}

.text {
  border: none;
  width: 89%;
  padding: 15px 20px;
  display: block;
  height: 15
    px;
  border-radius: 15px;
  background: #fff;
  border: 1px solid #6322EE;
  overflow: hidden;
  margin-top: 15px;
  transition: all 0.5s ease-in-out;
}

.text:focus {
  outline: 0;
  border: 1px solid #6322EE;
  border-radius: 15px;
  background: #fff;
}

.text:focus + span {
  opacity: 0.6;
}

input[type="text"],
input[type="password"] {
  font-family: 'Montserrat', sans-serif;
  color: #111;
}



input {
  display: inline-block;
  padding-top: 20px;
  font-size: 14px;
}

h2,
.signin {
  background-color: #AA86F6;
  color: #FFF;
  width: 100%;
  padding: 10px 20px;
  display: block;
  height: 39px;
  border-radius: 15px;
  margin-top: 30px;
  transition: all 0.5s ease-in-out;
  border: none;
  text-transform: uppercase;
```

```
}

.signin:hover {
  background: #6322EE;
  cursor: pointer;
}

.signin:focus {
  outline: none;
}
```