МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації і комп'ютерних систем імені проф. І.В. Ельперіна
Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Андрій ФОРСЮК	_____ Сергій ГРИБКОВ
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
«08» грудня 2025р.	«08» грудня 2025р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності 122 Комп'ютерні науки
(код та назва спеціальності)
освітньо-професійної програми Управління інформацією та аналітика даних
на тему: Інформаційно-аналітична система аналізу ефективності веб-ресурсів на основі A/B тестування

Виконав: здобувач 2 курсу, групи КН-2-2М

Мельничук Роман Васильович _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник Костіков Микола Павлович _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти _____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач _____
(підпис)

Київ – 2025р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації і комп'ютерних систем імені проф. І.В. Ельперіна
Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки
Освітній ступінь магістр
Спеціальність 122 Комп'ютерні науки
(код і назва)
Освітньо-професійна програма Управління інформацією та аналітика даних
(назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних технологій, штучного інтелекту і кібербезпеки

_____ Сергій ГРИБКОВ
«05» листопада 2025 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Мельничука Романа Васильовича
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційно-аналітична система аналізу ефективності веб-ресурсів на основі A/B тестування
керівник роботи Костіков Микола Павлович, доцент, к.т.н,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «05» листопада 2025 року №906-кс
2. Строк подання здобувачем роботи «01» грудня 2025 року
3. Вихідні дані до роботи *дані про організаційну структуру підприємства ФОП «Горенко Тарас Анатолійович»; технології, які використовує підприємство; стан автоматизації на підприємстві.*
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Розділ 1. Аналіз діяльності підприємства та постановка задачі дослідження
Розділ 2. Дослідження та обґрунтування технологій, методів і алгоритмів для a/b тестування веб-ресурсів
Розділ 3. Реалізація та апробація інформаційно-аналітичної системи
5. Перелік графічного матеріалу
1) Схематичні зображення архітектури системи
2) Моделі даних і процесів
3) Елементи інтерфейсу адміністративної панелі
4) Матеріали, пов'язані з роботою візуального редактора
5) Графічні матеріали, що відображають результати аналітики

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Костіков М.П., доцент, к.т.н.	01.10.2025	15.10.2025
2	Костіков М.П., доцент, к.т.н.	15.10.2025	15.11.2025
3	Костіков М.П., доцент, к.т.н.	15.11.2025	30.11.2025

7. Дата видачі завдання: 01 жовтня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Сучасний стан інформаційно-аналітичних систем у сфері послуг	01.10.2025	Виконано
2	Недоліки та проблеми впровадження готових систем	15.10.2025	Виконано
3	Аналіз діяльності підприємства	02.11.2025	Виконано
4	Постановка задачі та розробка вимог	15.11.2025	Виконано
5	Проєктування та реалізація інтерфейсу користувача. Створення аналітичних інструментів. Опис та реалізація модулів системи. Розробка коду програми.	16.11.2025	Виконано
6	Підготовка пояснювальної записки, презентації	30.11.2025	Виконано

Здобувач _____ Роман МЕЛЬНИЧУК
(підпис) (ім'я та прізвище)

Керівник роботи _____ Микола КОСТІКОВ
(підпис) (ім'я та прізвище)

АНОТАЦІЯ

У кваліфікаційній роботі на тему «Розроблення інформаційно-аналітичної системи A/B-тестування для оцінювання ефективності інтерфейсних рішень веб-ресурсів» розглянуто теоретичні та прикладні аспекти застосування методів A/B-тестування в задачах оптимізації інтерфейсу веб-ресурсів. Сформульовано вимоги до інформаційно-аналітичної системи, розроблено її загальну архітектуру, описано структуру сховища даних та принципи функціонування серверної та клієнтської частин. Представлено реалізацію веб-інтерфейсу адміністративної панелі та візуального редактора, який забезпечує можливість формування варіантів інтерфейсу без залучення програмного коду. Здійснено апробацію розробленої системи на прикладі експерименту з оптимізації кнопки «Додати в кошик» та виконано аналіз отриманих даних на основі побудованої користувацької воронки.

Отримані результати підтверджують працездатність і ефективність запропонованої системи та її придатність для використання підприємствами з метою підвищення результативності веб-ресурсів, покращення користувацького досвіду та обґрунтованого прийняття управлінських рішень.

Ключові слова: A/B-ТЕСТУВАННЯ, ІНФОРМАЦІЙНО-АНАЛІТИЧНА СИСТЕМА, ВЕБ-АНАЛІТИКА, КОРИСТУВАЦЬКА ВОРОНКА, ЕКСПЕРИМЕНТАЛЬНИЙ МЕТОД, UX-ОПТИМІЗАЦІЯ, СХОВИЩЕ ДАНИХ, SDK.

ANNOTATION

The thesis entitled “Development of an Information-Analytical A/B Testing System for Evaluating the Effectiveness of Web Interface Solutions” examines the theoretical and applied aspects of using A/B testing methods in tasks related to optimizing the performance of web interfaces. The requirements for the information-analytical system were formulated, its overall architecture was developed, and the structure of the data storage as well as the principles of functioning of the server- and client-side components were described. The implementation of a web-based administrative panel and a visual editor, which enables the creation of interface variants without modifying the program code, is presented. The developed system was tested on an experiment focused on optimizing the “Add to Cart” button, and the collected data were analyzed using a constructed user interaction funnel.

The obtained results confirm the functionality and effectiveness of the proposed system and demonstrate its suitability for use by enterprises aiming to improve the performance of their web resources, enhance user experience, and make data-driven management decisions.

Keywords: A/B TESTING, INFORMATION-ANALYTICAL SYSTEM, WEB ANALYTICS, USER FUNNEL, EXPERIMENTAL METHOD, UX OPTIMIZATION, DATA STORAGE, SDK.

ЗМІСТ

ВСТУП

Актуальність теми зумовлена зростанням ролі веб-ресурсів у забезпеченні конкурентоспроможності компаній у цифровій економіці. Для підприємств, що розробляють та супроводжують веб-сайти клієнтів (інтернет-магазини, лендінги, сервісні платформи), ефективність інтерфейсу та користувацьких сценаріїв безпосередньо впливає на конверсію і фінансові результати замовників. Традиційні підходи, що ґрунтуються лише на загальних показниках веб-аналітики або експертних оцінках, не забезпечують достатньої обґрунтованості рішень щодо змін на сайті.

A/B тестування дає змогу порівнювати альтернативні варіанти інтерфейсу та функціоналу за статистично значущими метриками. Водночас наявні комерційні платформи A/B тестування часто є дорогими, складними для інтеграції та надлишковими для невеликих студій і фахівців, які працюють із клієнтськими проєктами. Для ФОП «Горенко Тарас Анатолійович», що спеціалізується на розробці та супроводі веб-ресурсів, це обумовлює потребу у власній спеціалізованій інформаційно-аналітичній системі A/B тестування, орієнтованій насамперед на підвищення ефективності веб-ресурсів клієнтів.

Метою кваліфікаційної роботи є розробка та дослідження інформаційно-аналітичної системи для підвищення ефективності веб-ресурсів клієнтів на основі A/B тестування, подієво-орієнтованого збору даних та аналізу воронок конверсії в умовах діяльності ФОП «Горенко Тарас Анатолійович».

Для досягнення мети необхідно розв’язати такі завдання:

1. проаналізувати діяльність базового підприємства та існуючі підходи до оцінювання ефективності веб-ресурсів клієнтів;
2. дослідити сучасні інструменти A/B тестування та веб-аналітики, визначити їх придатність для використання у клієнтських проєктах підприємства;
3. формалізувати основні задачі A/B тестування та аналізу воронок конверсії для веб-ресурсів;
4. обґрунтувати вибір технологій, архітектурних рішень та статистичних методів для побудови інформаційно-аналітичної системи;

5. спроєктувати архітектуру системи, структуру сховища подій і ключові програмні компоненти;
6. реалізувати прототип системи та інтегрувати його з веб-ресурсами клієнтів підприємства;
7. провести апробацію розробленої системи та оцінити її вплив на можливості підвищення ефективності веб-ресурсів.

Об’єктом дослідження є процес оцінювання ефективності веб-ресурсів у цифровому середовищі.

Предметом дослідження є моделі, методи, алгоритми та програмні засоби A/B тестування і аналізу воронок конверсії, реалізовані в спеціалізованій інформаційно-аналітичній системі.

У роботі використовуються методи аналізу та синтезу, функціонального та архітектурного моделювання, математичного і статистичного моделювання A/B експериментів, а також методи проєктування програмного забезпечення та експериментальної перевірки.

Наукова новизна: розроблено формальні моделі та алгоритми для інтегрованої архітектури системи A/B-тестування, яка включає детермінований розподіл користувачів на основі хешування та охоплює подієво-орієнтовану обробку даних для автоматизованого аналізу воронок конверсії та статистичної оцінки значущості результатів. Розробку адаптовано до обчислювальних обмежень невеликих підприємств у задачах веб-аналітики та оптимізації інтерфейсів.

Практичне значення результатів полягає у можливості застосування розробленої системи у діяльності ФОП «Горенко Тарас Анатолійович» для планування та аналізу A/B експериментів на веб-ресурсах клієнтів, що дозволяє підвищувати їхню ефективність і якість прийняття рішень щодо змін на сайті.

Апробація результатів здійснена в межах XII Міжнародної науково-технічної Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (27 листопада 2025 р., НУХТ), де у співавторстві

з М.П. Костіковим представлені тези «Підвищення ефективності веб-ресурсів шляхом застосування інтелектуальних методів аналізу даних в системах A/B тестування».

Кваліфікаційна робота виконувалась згідно із планом науково-дослідних робіт кафедри інформаційних технологій, штучного інтелекту і кібербезпеки Національного університету харчових технологій: НДР «Дослідження та використання сучасних інформаційних технологій для виконання функцій та завдань виробничого і організаційного управління підприємств харчової галузі» № ДР 0120U105386, 2020–2025 рр.

Кваліфікаційна робота викладена на 103 сторінках, містить 14 таблиць, 21 рисунок та 28 використаних джерел та включає вступ, три розділи, висновки, список використаних джерел та додатки.

РОЗДІЛ 1. АНАЛІЗ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Загальна характеристика ФОП «Горенко Тарас Анатолійович»

ФОП «Горенко Тарас Анатолійович» є підприємством, що працює у сфері інформаційних технологій та спеціалізується на розробці, супроводі й оптимізації веб-ресурсів клієнтів. Основний фокус діяльності підприємства пов’язаний із створенням і розвитком комерційних веб-проєктів — інтернет-магазинів, лендінгів та сервісних сайтів, орієнтованих на досягнення конкретних бізнес-цілей замовників (залучення лідів, збільшення обсягу продажів, підвищення конверсії тощо).

Підприємство було засновано як невеликий стартап, що на початковому етапі об’єднував кількох фахівців у галузі веб-дизайну та програмування. Від самого початку його розвиток орієнтувався не лише на виконання типових проєктів, а й на впровадження інноваційних підходів до побудови інтерфейсів, організації користувацьких сценаріїв та використання аналітики даних для прийняття рішень. Завдяки поєднанню технічної компетентності, гнучкості в роботі з клієнтами та увазі до результатів їхнього бізнесу підприємству вдалося закріпитися на ринку послуг у сфері веб-розробки, сформувати стабільну клієнтську базу та поступово розширити спектр послуг.

На поточному етапі ФОП «Горенко Тарас Анатолійович» функціонує як команда, до складу якої входять CEO, менеджер проєктів, дизайнери, програмісти, тестувальники, SEO-спеціаліст та бухгалтер. Така структура дає змогу охоплювати повний цикл робіт над клієнтськими веб-проєктами — від опрацювання вимог і проєктування інтерфейсу до розробки, тестування, впровадження, технічної підтримки й подальшої оптимізації на основі аналітичних даних.

Основні напрямки діяльності підприємства включають:

- розробку веб-сайтів «під ключ» для малого та середнього бізнесу, зокрема інтернет-магазинів та лендінгів;

- вдосконалення існуючих веб-ресурсів клієнтів (редизайн, рефакторинг, оптимізація продуктивності);
- SEO-супровід та роботи, спрямовані на покращення видимості сайтів у пошукових системах;
- впровадження та налаштування систем веб-аналітики, відстеження ключових показників ефективності;
- консультаційний супровід щодо покращення користувацького досвіду та конверсійних показників.

Суттєвою особливістю підприємства є орієнтація на довгострокову співпрацю з клієнтами. Це передбачає не лише створення веб-ресурсу, але й його подальший розвиток, включаючи експерименти з різними варіантами інтерфейсу, контенту та механік взаємодії з користувачами. У такому контексті особливого значення набуває можливість системно проводити A/B тестування та аналізувати поведінку користувачів на основі подієво-орієнтованих даних. Саме потреба забезпечити клієнтам не тільки технічно якісний веб-продукт, а й інструменти для його постійного вдосконалення зумовлює актуальність розробки власної інформаційно-аналітичної системи A/B тестування.

Загалом ФОП «Горенко Тарас Анатолійович» можна охарактеризувати як гнучке підприємство, орієнтоване на проєктну діяльність і роботу з клієнтськими веб-ресурсами, де ключовим пріоритетом є вимірюваний результат для замовника. Такий профіль діяльності визначає специфіку вимог до інформаційних систем підприємства й обґрунтовує необхідність подальшого аналізу його організаційної структури та інформаційних потоків, що буде розглянуто в підпункті 1.2.

1.2. Організаційна структура підприємства та інформаційні потоки

Організаційна структура ФОП «Горенко Тарас Анатолійович» є компактною та орієнтованою на проєктну діяльність. Управління підприємством здійснює керівник (CEO), який відповідає за стратегічний розвиток, формування

портфеля клієнтів, укладання договорів та контроль ключових рішень щодо розвитку сервісів і внутрішніх інструментів.

Операційне управління реалізується через менеджера проєктів, який координує роботу команди, взаємодіє із замовниками, формує та підтримує беклог задач, контролює терміни виконання робіт і відповідає за узгодження результатів із клієнтом. Безпосереднє створення та вдосконалення веб-ресурсів клієнтів забезпечують дизайнери (UI/UX), розробники (front-end та back-end), тестувальники (QA) та SEO-спеціаліст. Бухгалтер здійснює облік фінансових операцій, розрахунки із замовниками та підрядниками, а також виконує функції податкової звітності.

Організаційну структуру підприємства подано на рисунку 1.1.

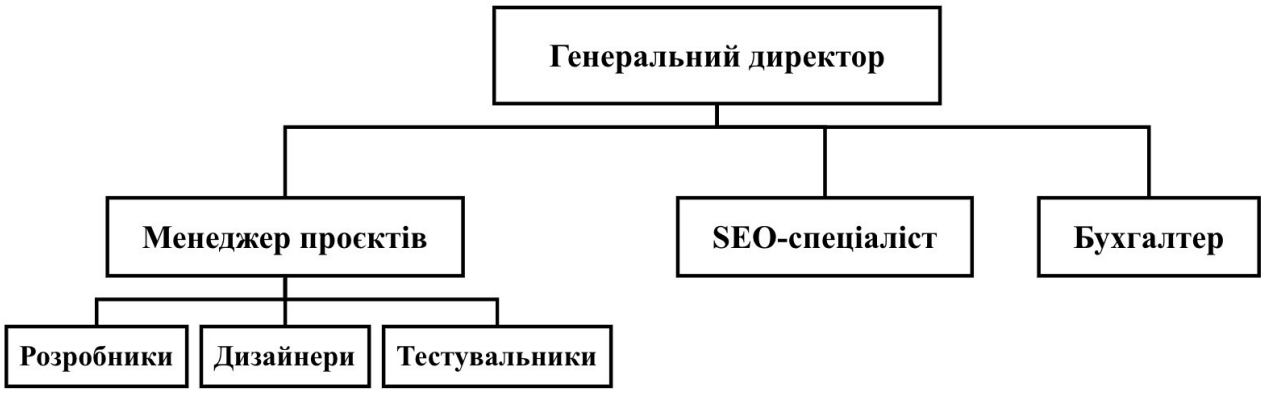


Рисунок 1.1 — Схема організаційної структури
ФОП «Горенко Тарас Анатолійович»

На рисунку 1.1 керівник підприємства розташований на верхньому рівні й безпосередньо взаємодіє з менеджером проєктів. Менеджер проєктів координує роботу дизайнерів, розробників, тестувальників та SEO-спеціаліста, тоді як бухгалтер підпорядковується керівнику та взаємодіє з менеджером проєктів з питань фінансового планування й звітності.

Організаційна структура визначає основні інформаційні потоки підприємства. На вхід до системи управління надходить інформація від клієнтів: бізнес-вимоги, побажання щодо функціоналу та дизайну, маркетингові цілі, дані про цільову аудиторію та обмеження за бюджетом і строками. Ця інформація

консолідується у менеджера проєктів і керівника, які трансформують її в цілі проєкту, технічні та організаційні завдання.

Дизайнери отримують від менеджера проєктів узгоджені вимоги та створюють прототипи, макети інтерфейсів і дизайн-системи, які передаються розробникам. Розробники, спираючись на макети, технічні завдання та рекомендації SEO-спеціаліста, реалізують функціонал веб-ресурсів клієнтів. Тестувальники одержують збірки програмного забезпечення від розробників і повертають результати тестування у вигляді звітів про дефекти та рекомендацій щодо покращення якості. SEO-спеціаліст формує вимоги до структури сайту, контенту, швидкодії, аналітики, а також інтерпретує дані веб-аналітики, результати A/B експериментів і показники воронок конверсії. Керівник отримує агреговану інформацію про стан проєктів, завантаженість команди та результати для клієнтів.

Основні ролі та відповідні інформаційні потоки узагальнено в таблиці 1.1.

Таблиця 1.1 — Основні ролі та інформаційні потоки у ФОП «Горенко Т.А.»

Роль	**Основні функції**	**Вхідна інформація**	**Вихідна інформація**
Керівник (CEO)	Стратегічне управління, робота з ключовими клієнтами, ухвалення рішень	Інформація про ринок, звіти менеджера проєктів, фінансові показники, відгуки клієнтів	Стратегічні цілі, затверджені бюджети, рішення щодо розвитку сервісів та внутрішніх інструментів
Менеджер проєктів	Координація команди, планування робіт, комунікація з клієнтами	Вимоги клієнтів, цілі проєкту, звіти від дизайнерів, розробників, тестувальників, SEO-спеціаліста	Технічні завдання, план-графіки, пріоритизація задач, звіти для керівника й клієнтів

Продовження таблиці 1.1

Роль	**Основні функції**	**Вхідна інформація**	**Вихідна інформація**
Дизайнер (UI/UX)	Проєктування інтерфейсів, підготовка макетів і прототипів	Вимоги клієнта, бриф, технічні обмеження, аналітика поведінки користувачів	Макети інтерфейсів, прототипи, дизайн-специфікації
Розробник (Front/Back-end)	Реалізація функціоналу, інтеграція сервісів, підтримка працездатності	Макети, технічні завдання, рекомендації SEO-спеціаліста, звіти про дефекти	Програмний код, реалізований функціонал, проміжні та фінальні збірки продукту
Тестувальник (QA)	Перевірка якості, пошук дефектів, контроль відповідності вимогам	Збірки продукту, технічні завдання, тестові сценарії	Звіти про дефекти, рекомендації щодо виправлення помилок та покращення стабільності
SEO-спеціаліст	Оптимізація веб-ресурсів, аналіз трафіку та конверсій, формування гіпотез	Дані веб-аналітики, результати A/B експериментів, інформація про цілі клієнта, технічні обмеження	Рекомендації щодо структури та контенту, налаштування аналітики

Закінчення таблиці 1.1

Роль	**Основні функції**	**Вхідна інформація**	**Вихідна інформація**
Бухгалтер	Облік фінансів, розрахунки з клієнтами, податкова звітність	Договори, акти виконаних робіт, платіжні документи, дані про витрати	Фінансова звітність, рахунки, акти, інформація про фінансові результати

З погляду подальшого впровадження інформаційно-аналітичної системи A/B тестування особливого значення набувають інформаційні потоки між менеджером проєктів, розробниками, SEO-спеціалістом та клієнтами. Саме ці потоки забезпечують формування гіпотез, вибір цільових метрик, планування експериментів, інтерпретацію результатів і прийняття рішень щодо змін на веб-ресурсах клієнтів. У подальших підпунктах буде розглянуто, яким чином такі потоки підтримуються наявними інформаційними системами підприємства та чому виникає потреба в спеціалізованій системі A/B тестування та аналізу ефективності веб-ресурсів.

Таким чином, організаційна структура ФОП «Горенко Тарас Анатолійович» забезпечує чіткий розподіл функцій між учасниками проєктів і формує систему інформаційних потоків, орієнтовану на створення й розвиток клієнтських веб-ресурсів, що визначає вимоги до подальшої автоматизації процесів аналізу їх ефективності.

1.3. Аналіз поточного стану автоматизації та використовуваних інформаційних систем

Рівень автоматизації бізнес-процесів є важливим чинником ефективності роботи ФОП «Горенко Тарас Анатолійович», оскільки підприємство реалізує повний цикл робіт із розробки та оптимізації веб-ресурсів клієнтів. Для підтримки цих процесів використовується низка сучасних інформаційних систем, які

забезпечують планування робіт, спільну розробку, комунікацію та ведення документації.

Основні інформаційні системи, що застосовуються на підприємстві, наведено в таблиці 1.2.

Таблиця 1.2 — Основні інформаційні системи, що використовуються на підприємстві

Система	**Призначення**	**Роль у клієнтських проєктах**
Jira	Управління проєктами та завданнями	Планування робіт, постановка задач, контроль строків і прогресу
Git, GitHub	Контроль версій та спільна розробка	Зберігання коду, ведення історії змін, організація командної роботи
GitHub Actions	Безперервна інтеграція та доставка (CI/CD)	Автоматизована збірка, тестування та розгортання веб-проєктів
Slack	Внутрішня комунікація	Оперативний обмін повідомленнями, координація роботи над проєктами
Google Drive	Хмарне сховище та документообіг	Зберігання договорів, брифів, макетів, технічної документації
Notion	Управління знаннями та внутрішньою документацією	Ведення технічних специфікацій, інструкцій, баз знань за проєктами

Зазначені системи забезпечують упорядковане ведення клієнтських проєктів, прозорість виконання робіт та зручний доступ до необхідної інформації для всіх учасників команди.

Разом із тим, наявні інформаційні системи орієнтовані переважно на підтримку процесів розробки, управління задачами та документообіг. Вони не забезпечують цілісного циклу роботи з даними про поведінку користувачів на веб-ресурсах клієнтів, зокрема збору, зберігання та аналізу результатів A/B тестування й воронок конверсії. Окремі функції веб-аналітики реалізуються за

допомогою зовнішніх сервісів, однак вони не інтегровані у власну інфраструктуру підприємства й не дозволяють уніфікувати підхід до оцінювання ефективності клієнтських веб-ресурсів.

Відсутність спеціалізованої інформаційно-аналітичної системи A/B тестування обмежує можливості підприємства щодо систематичного підвищення результативності клієнтських сайтів на основі об’єктивних даних. Це зумовлює необхідність подальшого аналізу існуючих рішень у галузі A/B тестування та веб-аналітики (підпункт 1.5) і постановки задачі розробки власної системи, орієнтованої на потреби ФОП «Горенко Тарас Анатолійович» та його клієнтів.

Таким чином, поточний стан автоматизації можна охарактеризувати як достатній для ефективної організації розробки, але недостатній щодо комплексної підтримки процесів аналізу ефективності клієнтських веб-ресурсів, що формує передумови для подальшого розвитку інформаційної інфраструктури підприємства.

1.4. Функціональне моделювання бізнес-процесів аналізу ефективності веб-ресурсів

Аналіз ефективності веб-ресурсів клієнтів у діяльності ФОП «Горенко Тарас Анатолійович» реалізується як послідовність взаємопов’язаних бізнес-процесів: від постановки цілей оптимізації до впровадження змін на сайтах клієнтів за результатами A/B тестування. Для формалізації цих процесів використано нотацію BPMN, що дозволяє чітко описати учасників, події, потоки робіт і точки ухвалення рішень, а також визначити вимоги до інформаційно-аналітичної системи A/B тестування.

Першим ключовим процесом є онбординг користувача та налаштування проєкту в системі. На цьому етапі створюється обліковий запис, здійснюється автентифікація, формується новий проєкт, який відповідає конкретному веб-ресурсу клієнта. В межах процесу задаються основні параметри проєкту: назва, домен сайту, тип веб-ресурсу (інтернет-магазин, лендінг тощо), цільові показники

ефективності та базові налаштування відслідковуваних подій. Узагальнену BPMN-діаграму цього процесу наведено на рисунку А.1 (Додаток А).

Другим важливим процесом є створення та налаштування A/B експерименту. Він містить послідовність дій: вибір проєкту, формулювання гіпотези, визначення контрольного та тестових варіантів, задання часток розподілу трафіку між варіантами, вибір цільових метрик (конверсія, кліки, досягнення певних подій), а також перевірку коректності налаштувань перед запуском. У випадку виявлення помилок або неповних даних система повертає користувача до відповідних кроків конфігурації. Структуру цього бізнес-процесу подано на рисунку А.2 (Додаток А).

Подальший етап пов’язаний із технічною інтеграцією клієнтського веб-ресурсу з інформаційно-аналітичною системою A/B тестування. Для цього використовується клієнтський SDK, який ініціалізується на веб-сторінці, отримує з серверної частини список активних експериментів та їх параметри, виконує розподіл користувачів між варіантами (bucket-процес) на основі детермінованого алгоритму та забезпечує відображення відповідного варіанта інтерфейсу. У цьому ж процесі SDK реєструє події (перегляди сторінок, кліки, досягнення цілей) і надсилає їх до сховища подій. Узагальнену BPMN-діаграму цього процесу наведено на рисунку А.3 (Додаток А).

Зібрані події використовуються для побудови воронок конверсії та розрахунку результатів A/B експериментів. На цьому етапі відбувається агрегація подій за користувачами, сесіями, варіантами експерименту та кроками воронки. Система обчислює ключові показники (коефіцієнти конверсії, відносне покращення, статистичну значущість різниць між варіантами), формує візуальні подання (графіки, діаграми воронок) і надає їх у панелі адміністратора. SEO-спеціаліст та менеджер проєктів, аналізуючи ці дані, приймають рішення щодо впровадження виграшного варіанта, коригування гіпотез або запуску нових експериментів. Структуру бізнес-процесу аналізу результатів A/B тестування та воронок конверсії подано на рисунку А.4 (Додаток А).

Сукупність наведених BPMN-діаграм відображає безперервний цикл роботи з оптимізацією клієнтських веб-ресурсів: від первинного налаштування проєкту та формування гіпотез до збору даних, аналізу результатів і ухвалення рішень щодо змін на сайтах. Така формалізація дозволяє чітко визначити точки інтеграції майбутньої інформаційно-аналітичної системи з клієнтськими веб-ресурсами та внутрішніми процесами підприємства, а також сформувати вимоги до функціоналу системи на рівні підтримки кожного з описаних бізнес-процесів.

Таким чином, функціональне моделювання бізнес-процесів аналізу ефективності веб-ресурсів забезпечує структуроване уявлення про роботу підприємства в цьому напрямі та слугує основою для подальшого огляду існуючих технологій і інструментів A/B тестування та веб-аналітики, що розглядатиметься в підпункті 1.5.

1.5. Огляд існуючих технологій, інструментів та інформаційних систем для A/B тестування і веб-аналітики

Для підвищення ефективності веб-ресурсів на ринку представлено значну кількість технологій та сервісів, що підтримують проведення A/B тестування, мультиваріантних експериментів, аналіз поведінки користувачів та побудову воронок конверсії. Ці рішення можна умовно поділити на дві групи: спеціалізовані платформи A/B тестування та інструменти продуктової аналітики, а також системи загальної веб-аналітики.

До спеціалізованих систем A/B тестування належать, зокрема, Optimizely, VWO, AB Tasty, Adobe Target та подібні сервіси. Вони надають розвинений функціонал для налаштування експериментів, гнучких сценаріїв таргетингу, сегментації користувачів, а також інструменти візуального редагування інтерфейсів і побудови складних експериментальних схем. Такі платформи орієнтовані переважно на компанії середнього та великого бізнесу, мають широкий спектр інтеграцій з іншими сервісами (CRM, CDP, аналітичні платформи) та розвинені засоби безпеки й керування доступом.

Разом з тим, використання подібних систем для невеликих студій та фахівців, що працюють із клієнтськими веб-ресурсами, пов’язане з низкою обмежень: високою вартістю підписок, складністю інтеграції в наявні процеси та технічну інфраструктуру, а також залежністю від зовнішнього постачальника в частині зберігання та обробки даних. Для проєктів малого та середнього бізнесу це часто створює надмірний бар’єр входу, тоді як обсяг необхідного функціоналу є значно скромнішим, ніж можливості корпоративних платформ.

Основні характеристики типових систем A/B тестування узагальнено в таблиці 1.3.

Таблиця 1.3 — Узагальнена характеристика типових систем A/B тестування

Характеристика	**Корпоративні платформи (Optimizely, VWO, AB Tasty тощо)**	**Потреби підприємств типу ФОП «Горенко Т.А.»**
Цільова аудиторія	Середні та великі компанії, корпорації	Малі та середні бізнеси, проєктні студії, окремі фахівці
Модель доступу	SaaS з підпискою, багаторівневі тарифні плани	Гнучка, бажано власна система з контролем витрат
Функціонал A/B та мультиваріантних тестів	Розширений, включаючи складні сценарії таргетингу та персоналізації	Базовий, орієнтований на типові експерименти з інтерфейсом і контентом
Інтеграція з іншими системами	Широкий набір інтеграцій (CRM, CDP, аналітика, рекламі платформи)	Інтеграція насамперед із фронтендом клієнтських сайтів та базовою аналітикою

Закінчення таблиці 1.3

Характеристика	**Корпоративні платформи (Optimizely, VWO, AB Tasty тощо)**	**Потреби підприємств типу ФОП «Горенко Т.А.»**
Зберігання та обробка даних	У хмарній інфраструктурі постачальника	Бажаний контроль над даними (власне сховище подій, можливість гнучкої обробки)
Поріг входу за вартістю та складністю	Високий для невеликих команд	Має бути низьким, з урахуванням ресурсів невеликого підприємства

Окрему групу становлять інструменти веб-аналітики загального призначення (наприклад, Google Analytics, Mixpanel, Amplitude, Matomo та ін.). Вони забезпечують збір і агрегування подій, аналіз трафіку, побудову сегментів аудиторії, воронок конверсії та ряд базових або розширених звітів. Частина таких систем має вбудовані можливості для простих експериментів (A/B або редірект-тести), проте основний фокус зосереджено на аналізі поведінки користувачів, а не на системному управлінні експериментами.

У практиці використання подібних інструментів для клієнтських веб-ресурсів виникають такі особливості:

- налаштування відстеження подій і воронок часто виконується вручну для кожного проєкту;
- відсутня єдина модель даних, що ускладнює порівняння результатів між різними клієнтськими сайтами;
- інтерфейси орієнтовані на широке коло користувачів, але не завжди зручні для щоденної роботи невеликої технічної команди, яка поєднує функції розробки та аналітики;
- результати A/B експериментів (якщо вони проводяться) не завжди інтегровані з внутрішніми процесами ухвалення рішень щодо змін на сайті.

Для ФОП «Горенко Тарас Анатолійович», який супроводжує веб-ресурси клієнтів на всіх етапах життєвого циклу, ключовою вимогою є не лише наявність аналітичних звітів, а й можливість будувати єдиний цикл: постановка гіпотези — налаштування експерименту — автоматизований збір подій — розрахунок показників — інтерпретація результатів — впровадження змін. Існуючі окремі інструменти A/B тестування та веб-аналітики частково покривають ці етапи, але, як правило, не пропонують компактного, адаптованого до невеликої студії інтегрованого рішення з єдиним сховищем подій і спеціалізованим інтерфейсом для роботи з клієнтськими проєктами.

Таким чином, аналіз існуючих технологій і інструментів показує, що, попри наявність потужних корпоративних платформ A/B тестування та гнучких систем веб-аналітики, для підприємств масштабу ФОП «Горенко Тарас Анатолійович» залишається актуальною задача створення власної інформаційно-аналітичної системи. Така система має поєднувати достатній для повсякденної практики функціонал A/B тестування та аналізу воронок з контролем над даними, адаптацією до специфіки клієнтських веб-ресурсів і невисоким порогом входу за вартістю та складністю використання. Це обґрунтовує необхідність подальшого дослідження методів і технологій, які будуть покладені в основу проєктованої системи.

1.6. Аналітичний огляд наукових підходів до A/B тестування та оцінювання ефективності веб-ресурсів

У наукових та прикладних дослідженнях оцінювання ефективності веб-ресурсів розглядається як багатофакторна задача, що поєднує методи веб-аналітики, статистичного аналізу, експериментального дизайну та оптимізації користувацького досвіду. Значна увага приділяється саме експериментальним підходам, серед яких A/B тестування є базовим інструментом перевірки гіпотез щодо впливу змін інтерфейсу, контенту або бізнес-логіки на поведінку користувачів та ключові показники ефективності.

У роботах, присвячених A/B тестуванню веб-інтерфейсів, наголошується на необхідності коректної постановки гіпотези, вибору адекватних метрик (коефіцієнт конверсії, клікрейти, показники доходу, час взаємодії тощо) та застосування статистично обґрунтованих критеріїв порівняння варіантів. Широко розглядаються проблеми похибок першого та другого роду, вплив розміру вибірки та тривалості експерименту на довірчість висновків, а також ризики «перегляду результатів у процесі» (peeking), які можуть призводити до хибних рішень.

Окремим напрямом є дослідження, присвячені оптимізації структури та контенту веб-ресурсів на основі аналізу воронок конверсії. У межах цього підходу користувацький шлях розглядається як послідовність кроків (подій), що ведуть до цільової дії, а задача аналізу полягає у виявленні «вузьких місць» — етапів із найбільшими втратами користувачів. Наукові праці в цій галузі описують методи побудови та аналізу воронок, моделі переходів між кроками, використання сегментації аудиторії та аналізу когорти для глибшого розуміння поведінки користувачів.

Значна кількість досліджень розглядає подієво-орієнтований підхід до збору даних, коли основою для аналізу є не агреговані показники за сесіями, а окремі події, що фіксують дії користувача у часі. У такому підході особливого значення набувають питання проєктування схеми подій, вибору ідентифікаторів користувача та сесії, забезпечення цілісності даних при роботі з розподіленими системами та високонавантаженими веб-ресурсами. Подієво-орієнтовані моделі даних створюють підґрунтя для реалізації більш гнучких аналітичних сценаріїв, включаючи побудову воронок, сегментацію, ретроспективний аналіз та оцінювання результатів експериментів.

У літературі також розвинутий напрям, пов’язаний із побудовою платформ експериментування та інтеграцією A/B тестування у процеси розробки цифрових продуктів. Такі роботи описують архітектурні підходи до побудови систем, що підтримують масштабоване проведення експериментів, управління життєвим циклом експериментів, централізоване зберігання аналітичних даних та

забезпечення відтворюваності результатів. Значний акцент робиться на забезпеченні детермінованого розподілу користувачів між варіантами, узгодженій інфраструктурі збору подій та стандартизованих підходах до обчислення метрик.

Для задач оцінювання ефективності веб-ресурсів клієнтів, характерних для ФОП «Горенко Тарас Анатолійович», у наукових підходах особливо релевантними є поєднання класичного A/B тестування з аналізом воронок конверсії та подієво-орієнтованим збором даних. Таке поєднання дозволяє не лише порівнювати загальні показники ефективності варіантів (наприклад, загальний коефіцієнт конверсії), а й аналізувати, на яких саме етапах користувацького шляху відбуваються покращення або погіршення, і за рахунок яких змін інтерфейсу чи сценаріїв взаємодії вони досягаються.

Узагальнюючи, аналітичний огляд наукових підходів показує, що сучасні дослідження в галузі A/B тестування та веб-аналітики концентруються навколо таких ключових ідей: використання експериментального дизайну для перевірки гіпотез, подієво-орієнтоване збирання даних, аналіз воронок конверсії, побудова платформ для масштабованих експериментів та інтеграція аналітики в процеси розробки цифрових продуктів. Ці ідеї закладають теоретичне підґрунтя для проєктування інформаційно-аналітичної системи, адаптованої до потреб підприємства, що працює з клієнтськими веб-ресурсами, і визначають подальші напрямки дослідження у межах даної роботи.

1.7. Виявлення проблеми, формулювання мети, об’єкта, предмета та задач дослідження

Аналіз діяльності ФОП «Горенко Тарас Анатолійович» показує, що підприємство орієнтоване на розробку, супровід та оптимізацію веб-ресурсів клієнтів, серед яких переважають інтернет-магазини, лендінги та сервісні сайти. Ключовим критерієм успішності таких проєктів є досягнення вимірюваних бізнес-результатів замовників: зростання кількості заявок, збільшення обсягу продажів, підвищення коефіцієнта конверсії, покращення показників залученості

користувачів. Це зумовлює потребу у системному підході до оцінювання ефективності веб-ресурсів на основі об’єктивних даних.

Проведений аналіз організаційної структури та стану автоматизації показав, що на підприємстві вже використовуються сучасні інструменти для управління проєктами, контролю версій, безперервної інтеграції, комунікацій та документообігу. Водночас спеціалізована підтримка A/B тестування та аналізу воронок конверсії реалізована лише фрагментарно, переважно за рахунок окремих зовнішніх сервісів веб-аналітики. Це ускладнює побудову єдиного експериментального контуру для клієнтських веб-ресурсів, веде до дублювання налаштувань та відсутності узгодженої моделі даних.

Огляд існуючих технологій та інструментів показав, що корпоративні платформи A/B тестування (Optimizely, VWO, AB Tasty, Adobe Target тощо) орієнтовані на великі компанії, мають високий поріг входу за вартістю й складністю інтеграції та не завжди доцільні для застосування у проєктах малого та середнього бізнесу. Системи загальної веб-аналітики забезпечують збір і базовий аналіз подій, але не пропонують спеціалізованої, інтегрованої для підприємства платформи експериментування, адаптованої до особливостей клієнтських проєктів ФОП «Горенко Тарас Анатолійович».

Таким чином, виявляється суперечність між потребою підприємства у систематичному проведенні A/B тестування та аналізу воронок конверсії для клієнтських веб-ресурсів і відсутністю власної інформаційно-аналітичної системи, яка б забезпечувала повний цикл роботи з експериментами на основі подієво-орієнтованих даних. Це дозволяє сформулювати основну проблему дослідження як задачу розробки та обґрунтування такої системи, що відповідала б масштабам і специфіці діяльності підприємства та його клієнтів.

Відповідно до виявленої проблеми мету дослідження сформульовано так: розробка та дослідження інформаційно-аналітичної системи для підвищення ефективності веб-ресурсів клієнтів на основі A/B тестування, подієво-орієнтованого збору даних та аналізу воронок конверсії в умовах діяльності ФОП «Горенко Тарас Анатолійович».

Об’єктом дослідження є процес оцінювання ефективності веб-ресурсів у цифровому середовищі.

Предметом дослідження є моделі, методи, алгоритми та програмні засоби A/B тестування й аналізу воронок конверсії, реалізовані в спеціалізованій інформаційно-аналітичній системі, призначеній для роботи з клієнтськими веб-ресурсами.

Для досягнення поставленої мети необхідно розв’язати такі задачі дослідження:

1. Проаналізувати діяльність базового підприємства, його організаційну структуру та інформаційні потоки, пов’язані з розробкою та оптимізацією веб-ресурсів клієнтів.
2. Оцінити поточний стан автоматизації на підприємстві та виявити обмеження наявних підходів до оцінювання ефективності клієнтських веб-ресурсів.
3. Провести огляд сучасних технологій, інструментів та інформаційних систем для A/B тестування і веб-аналітики, визначити їх придатність для застосування в умовах діяльності підприємства.
4. Формалізувати основні задачі A/B тестування та аналізу воронок конверсії для клієнтських веб-ресурсів на основі подієво-орієнтованого збору даних.
5. Обґрунтувати вибір технологічного стеку, архітектурних рішень і статистичних методів, що будуть покладені в основу інформаційно-аналітичної системи.
6. Спроєктувати архітектуру системи, структуру сховища подій та основні програмні компоненти (серверні сервіси, клієнтський SDK, модуль побудови воронок, панель адміністратора).
7. Реалізувати прототип інформаційно-аналітичної системи та інтегрувати його з веб-ресурсами клієнтів підприємства.
8. Провести апробацію розробленої системи на прикладі A/B експериментів для клієнтських веб-ресурсів та оцінити її вплив на можливості підвищення ефективності цих ресурсів.

Сформульовані об’єкт, предмет, мета та задачі дослідження узгоджуються з результатами аналізу діяльності підприємства та існуючих рішень і визначають структуру подальших розділів роботи. У другому розділі буде розглянуто теоретичні та методичні засади моделювання задач A/B тестування й аналізу воронок, а також обґрунтовано вибір технологій та методів, що використовуватимуться при побудові інформаційно-аналітичної системи.

1.8. Висновки до розділу 1

У першому розділі розглянуто діяльність ФОП «Горенко Тарас Анатолійович» як підприємства, що спеціалізується на розробці, супроводі та оптимізації веб-ресурсів клієнтів. Наведено загальну характеристику підприємства, визначено його основні напрями діяльності та акцент на досягненні вимірюваних бізнес-результатів для замовників.

Проаналізовано організаційну структуру підприємства та основні інформаційні потоки, що забезпечують взаємодію між керівником, менеджером проєктів, дизайнерами, розробниками, тестувальниками, SEO-спеціалістом і бухгалтером. Показано, що така структура орієнтована на проєктну роботу з клієнтськими веб-ресурсами й підтримує повний цикл їх розробки та розвитку.

Оцінено поточний стан автоматизації діяльності підприємства та визначено, що використовувані інформаційні системи ефективно підтримують управління проєктами, спільну розробку, комунікацію та документообіг. Водночас виявлено відсутність інтегрованої спеціалізованої системи для A/B тестування та аналізу воронок конверсії, що обмежує можливості систематичного підвищення ефективності клієнтських веб-ресурсів.

На основі огляду існуючих технологій, інструментів та інформаційних систем для A/B тестування й веб-аналітики показано, що корпоративні платформи мають високий поріг входу та не повною мірою відповідають потребам підприємств масштабу ФОП «Горенко Тарас Анатолійович», тоді як загальні системи веб-аналітики не забезпечують повного циклу роботи з експериментами. Це дозволило сформулювати проблему дослідження, визначити об’єкт, предмет,

мету та задачі роботи, спрямовані на розробку інформаційно-аналітичної системи, адаптованої до потреб підприємства.

Отримані результати створюють необхідне підґрунтя для переходу до теоретичного та методичного обґрунтування моделей, методів і алгоритмів, що будуть застосовані при побудові інформаційно-аналітичної системи A/B тестування, що є предметом другого розділу.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ ТА ОБҐРУНТУВАННЯ ТЕХНОЛОГІЙ, МЕТОДІВ І АЛГОРИТМІВ ДЛЯ A/B ТЕСТУВАННЯ ВЕБ-РЕСУРСІВ

2

2.1. Моделювання задачі оцінки ефективності веб-ресурсів на основі A/B тестування

Оцінювання ефективності веб-ресурсів клієнтів на основі A/B тестування розглядається як задача порівняння двох або більше варіантів інтерфейсу, контенту чи бізнес-логіки за сукупністю кількісних показників. У загальному випадку кожен варіант веб-ресурсу взаємодіє з певною множиною користувачів, для яких фіксуються події, що відображають їхню поведінку (перегляди сторінок, кліки, додавання товарів до кошика, оформлення замовлення тощо). На основі цих подій обчислюються показники ефективності, які далі використовуються для порівняння варіантів.

У межах моделі A/B тестування розглядаються щонайменше два варіанти: базовий (контрольний) та один або кілька тестових. Для кожного варіанта формується множина користувачів, які випадковим, але детермінованим чином (з урахуванням принципів sticky bucketing) закріплюються за певною альтернативою. Для кожного користувача фіксується факт виконання чи невиконання цільової дії, а також додаткові характеристики взаємодії з веб-ресурсом. Це дозволяє переходити від окремих подій до агрегованих показників на рівні варіантів.

Ключовим поняттям моделі є конверсія — відношення кількості користувачів, які виконали цільову дію (наприклад, оформлення замовлення, відправка форми, підписка), до загальної кількості користувачів, що взаємодіяли з відповідним варіантом веб-ресурсу. На основі конверсії визначаються похідні показники: абсолютна різниця між варіантами, відносне покращення (uplift), а також інші метрики, пов’язані з доходом, середнім чеком, глибиною перегляду та залученістю.

Для коректної постановки задачі оцінювання ефективності необхідно формалізувати:

- набір ключових метрик, що використовуються як цілі оптимізації (коефіцієнт конверсії, середній дохід на користувача, коефіцієнт переходу між кроками воронки тощо);
- схему розподілу користувачів між варіантами, яка забезпечує відтворюваність результатів і відсутність систематичних зміщень;
- подієво-орієнтовану модель даних, що описує структуру подій, ідентифікатори користувачів і сесій, а також атрибути, необхідні для побудови воронок;
- модель воронки конверсії, яка дозволяє описати послідовність кроків користувача та обчислити ймовірності переходів між ними.

У такій постановці задача оцінювання ефективності веб-ресурсів зводиться до побудови сукупності формальних моделей:

- метричної моделі, яка задає спосіб обчислення ключових показників для кожного варіанта;
- моделі призначення варіантів, що визначає алгоритм закріплення користувачів за варіантами експерименту;
- моделі даних подій, яка описує структуру сховища подій і зв’язки між ними;
- моделі воронок, яка описує послідовність подій, що веде до цільової дії, та дозволяє оцінювати втрати на кожному кроці.

2.1.1. Формалізація показників конверсії та ключових метрик ефективності

Для оцінювання ефективності веб-ресурсів у контексті A/B тестування вводиться система формальних показників, що дозволяє порівнювати варіанти інтерфейсу та робити обґрунтовані висновки щодо їх результативності.

Нехай для певного варіанта v маємо:

• N_v — кількість унікальних користувачів, які взаємодіяли з варіантом;

• C_v — кількість користувачів, які виконали цільову дію;

• E_v — загальна кількість цільових подій (наприклад, оформлень замовлень, відправок форм тощо);

• R_v — сумарний дохід, отриманий від користувачів варіанта v.

Базовим показником є коефіцієнт конверсії варіанта v, який визначається як відношення кількості користувачів, що виконали цільову дію, до загальної кількості користувачів, що побачили варіант:

$$CR_v = C_v / N_v \tag{2.1}$$

У випадку, коли одна й та сама цільова дія може виконуватися кілька разів одним користувачем (наприклад, кілька замовлень за сесію), доцільно розглядати інтенсивність цільових подій на одного користувача:

$$IR_v = E_v / N_v \tag{2.2}$$

де IR_v відображає середню кількість цільових подій на одного користувача для варіанта v.

Якщо веб-ресурс пов’язаний із монетизацією, важливим показником є середній дохід на користувача (ARPU) для варіанта v:

$$ARPU_v = R_v / N_v \tag{2.3}$$

Для порівняння варіантів A та B використовуються як абсолютні, так і відносні характеристики. Абсолютна різниця коефіцієнтів конверсії визначається як:

$$\Delta CR = CR_B - CR_A \tag{2.4}$$

де CR_A — конверсія контрольного варіанта, CR_B — конверсія тестового варіанта.

Відносне покращення (uplift) конверсії тестового варіанта B порівняно з контрольним A задається виразом:

$$U_{CR} = ((CR_B - CR_A) / CR_A) \cdot 100\% \quad (2.5)$$

Аналогічно можуть визначатися відносні зміни для інших показників (наприклад, ARPU, IR), що дає змогу оцінювати не лише ймовірність настання цільової події, а й її «вартість» для бізнесу клієнта.

У практиці оцінювання ефективності веб-ресурсів доцільно розглядати також проміжні метрики, пов’язані з окремими кроками воронки (перехід на сторінку товару, додавання до кошика, перехід до оформлення замовлення тощо). Для кожного кроку k можна ввести:

• $N_{v,k}$ — кількість користувачів, які досягли кроку k у варіанті v;

• $N_{v,k+1}$ — кількість користувачів, які перейшли з кроку k на крок k+1.

Тоді локальний коефіцієнт переходу між кроками воронки визначається як:

$$CR_{v,k \to k+1} = N_{v,k+1} / N_{v,k} \quad (2.6)$$

Цей показник використовується для виявлення етапів, на яких спостерігаються найбільші втрати користувачів, та для подальшого формування гіпотез щодо оптимізації.

Сукупність формально визначених метрик — CR_v, IR_v, $ARPU_v$, локальні коефіцієнти переходу у воронці та відповідні відносні зміни між варіантами — утворює основу метричної моделі оцінювання ефективності веб-ресурсів.

Подальший розвиток моделі пов’язаний із формалізацією способу, у який користувачі закріплюються за варіантами експерименту. У підпункті 2.1.2 буде розглянуто модель розподілу користувачів між варіантами із використанням детермінованого хешування та принципів sticky bucketing, що забезпечує відтворюваність результатів і коректність обчислення зазначених метрик.

2.1.2. Модель розподілу користувачів між варіантами експерименту (sticky bucketing, хешування)

Однією з ключових складових A/B тестування є механізм розподілу користувачів між варіантами експерименту. Від якості цього механізму залежать коректність обчислення метрик, відсутність систематичних зміщень та можливість інтерпретації результатів. Модель розподілу повинна забезпечувати:

- випадковий, але детермінований вибір варіанта для конкретного користувача;
- збереження вибраного варіанта для користувача протягом усього часу дії експерименту (sticky bucketing);
- відповідність фактичного розподілу користувачів заданим вагам варіантів (наприклад, 50/50, 70/30 тощо);
- незалежність експериментів один від одного.

Нехай для експерименту e задано множину варіантів $V_e = \{v_1, v_2, \ldots, v_m\}$ та відповідні ваги $w_{v_1}, w_{v_2}, \ldots, w_{v_m}$, де

$$\sum_{i=1}^{m} w_{v_i} = 1, w_{v_i} > 0. \quad (2.7)$$

Для кожного користувача визначається стійкий ідентифікатор *user_id*, який може базуватися на cookies, локальному сховищі браузера або інших механізмах. Для кожної пари «користувач–експеримент» обчислюється детермінована хеш-функція:

$$h = H(user\_id, experiment\_id), \quad (2.8)$$

де $H(\cdot)$ — хеш-функція, що повертає ціле число великого діапазону. Далі значення h нормується на відрізок $[0,1)$ наприклад:

$$u = \frac{h \bmod M}{M}, \tag{2.9}$$

де M — фіксоване ціле число (наприклад, $M = 10^6$), а u можна інтерпретувати як псевдовипадкове число, рівномірно розподілене на [0,1) для даної пари *user_id* та *experiment_id*.

Розподіл користувача між варіантами здійснюється шляхом розбиття інтервалу [0,1) на підінтервали, пропорційні вагам варіантів. Нехай обчислено кумулятивні суми ваг:

$$S_0 = 0, S_k = \sum_{i=1}^{k} w_{v_i}, k = 1,\ldots,m. \tag{2.10}$$

Тоді користувач призначається до варіанта v_k, для якого виконується нерівність

$$S_{k-1} \le u < S_k. \tag{2.11}$$

Такий підхід забезпечує:

- відтворюваність: для фіксованих *user_id*, *experiment_id* та конфігурації ваг користувач завжди потрапляє до одного й того самого варіанта;
- відповідність вагам: за достатньо великої кількості користувачів частка користувачів у кожному варіанті наближається до відповідної ваги w_{v_k}.

Sticky bucketing реалізується завдяки тому, що для кожної пари «користувач–експеримент» використовується одна й та сама хеш-функція та конфігурація розподілу. Додатково результат призначення може кешуватися на стороні клієнта (наприклад, у локальному сховищі браузера) та/або на сервері у таблиці призначень, що прискорює повторні звернення та зменшує навантаження на обчислення.

Для запуску кількох незалежних експериментів використовується окремий *experiment_id* для кожного з них. Це гарантує, що хеш-значення та, відповідно,

призначення варіантів не будуть корельованими між експериментами, якщо явно не передбачено інше.

Запропонована модель розподілу користувачів поєднує простоту реалізації, відтворюваність і контроль над частками трафіку, що робить її придатною для використання в інформаційно-аналітичній системі A/B тестування клієнтських веб-ресурсів. У наступному підпункті 2.1.3 буде розглянуто подієво-орієнтований підхід до збору даних та структуру подій, необхідних для розрахунку метрик ефективності та аналізу воронок конверсії.

2.1.3. Моделювання подієво-орієнтованого збору даних та структури подій

У проєктованій інформаційно-аналітичній системі оцінювання ефективності веб-ресурсів використовується подієво-орієнтований підхід до збору даних. Усі вимірювання поведінки користувачів базуються на послідовності атомарних подій, які реєструються клієнтським SDK у браузері та надсилаються на сервер за допомогою спеціального API методу POST /api/track. Подальші розрахунки метрик, побудова воронок конверсії та аналіз результатів A/B експериментів виконуються на основі цього потоку подій.

У системі виокремлюються основні типи подій, що описують користувацьку активність:

- View — перегляд сторінки або екрану інтерфейсу;
- Click — клік по відстежуваному елементу (кнопці, посиланню тощо);
- Conversion — досягнення цільової дії (наприклад, оформлення замовлення, відправка форми);
- Custom — кастомні бізнес-події, специфічні для конкретного веб-ресурсу клієнта.

Для забезпечення уніфікованого зберігання та обробки даних у системі запроваджено єдину структуру події. Кожна подія містить набір полів, необхідних для ідентифікації користувача, прив’язки до експерименту та варіанта, опису

контексту взаємодії й подальшого аналітичного опрацювання. Структуру події наведено в таблиці 2.1.

Таблиця 2.1 – Структура події, що надсилається з клієнтського SDK

Поле	**Опис**
user_id	Унікальний ідентифікатор користувача, сформований на стороні клієнта й збережений у браузері
session_id	Ідентифікатор сесії браузера
event_type	Тип події (наприклад, view, click, conversion, custom)
event_name	Кастомна назва події, що описує конкретну дію (наприклад, purchase_completed)
experiment_id	Ідентифікатор пов’язаного A/B експерименту
variant_id	Ідентифікатор варіанта, призначеного користувачу в межах експерименту
url	Поточна URL сторінки, на якій було зафіксовано подію
properties	Набір додаткових властивостей у форматі JSON (наприклад, вартість замовлення тощо)
timestamp	Час настання події
user_agent	Строка ідентифікації браузера
device_type	Тип пристрою (Mobile, Desktop, Tablet)
browser	Назва браузера
os	Операційна система
country	Географічне розташування користувача (якщо доступно)

Клієнтський SDK формує масив таких подій та періодично надсилає їх на сервер єдиним пакетом. На серверній стороні події валідуються, обробляються та зберігаються в аналітичному сховищі подій, оптимізованому для виконання запитів агрегування, фільтрації за часом, експериментом, варіантом та іншими атрибутами. На основі сирих подій надалі будуються агреговані представлення для обчислення показників A/B експериментів і аналізу воронок конверсії.

Подієво-орієнтована модель даних забезпечує:

- можливість точної прив’язки кожної дії користувача до конкретного експерименту та варіанта;
- гнучкість у визначенні цілей та кастомних бізнес-подій за рахунок поля *event_name* і структури *properties*;
- підтримку різнорівневого аналізу – від загальних агрегованих метрик до деталізованого розгляду окремих сегментів аудиторії за типом пристрою, браузером чи країною.

Саме на основі цієї структури подій реалізуються як розрахунок ключових метрик ефективності, описаних у підпункті 2.1.1, так і моделювання воронок конверсії, що дозволяє аналізувати послідовність дій користувача на шляху до цільової події. У підпункті 2.1.4 буде розглянуто формальну модель воронок конверсії, яка використовує події зазначеної структури для обчислення коефіцієнтів переходу між кроками та виявлення «вузьких місць» у користувацьких сценаріях.

2.1.4. Моделювання воронок конверсії на основі подієвих даних

Воронка конверсії у проєктованій системі розглядається як формальний опис багатокрокового шляху користувача до цільової дії на веб-ресурсі клієнта. Кожен крок воронки визначається за допомогою поєднання події SDK та умов фільтрації, а результати аналізу воронки використовуються для виявлення етапів із найбільшими втратами користувачів та оцінювання впливу A/B експериментів на поведінку.

Формально воронка F задається впорядкованою послідовністю кроків:

$$F = (s_1, s_2, \ldots, s_K), \tag{2.12}$$

де кожен крок s_K описується параметрами:

- назва події SDK (наприклад, page_view, button_click, purchase_completed);
- фільтр за URL (опціонально) — рядок для зіставлення;
- тип зіставлення (Contains, Exact, Regex).

Таким чином, крок s_k визначає підмножину подій з потоку подій, що відповідають певній дії користувача на конкретних сторінках або групі сторінок.

Для заданої воронки F та обраного інтервалу часу $[T_{start}, T_{end}]$, розглядається множина подій, збережених у сховищі подій (таблиця *events* у ClickHouse), що відповідають цьому діапазону. Кожна подія має, зокрема, такі атрибути: ідентифікатор користувача (*user_id*), ідентифікатор сесії (*session_id*), тип та назву події (*event_type*, *event_name*), URL сторінки (*url*), а також, за потреби, ідентифікатори експерименту та варіанта (*experiment_id*, *variant_id*), що дозволяє аналізувати воронки окремо для різних варіантів A/B експерименту.

Для кожного користувача визначається, чи існує в потоці його подій послідовність, що відповідає крокам воронки F у правильному порядку. Користувач вважається таким, що досяг кроку s_1, якщо в обраному інтервалі часу має хоча б одну подію, яка задовольняє умовам кроку. Аналогічно, користувач досягає кроку s_k, якщо після моменту досягнення кроку s_{k-1} у нього є подія, що відповідає кроку s_k. При цьому порядок подій визначається за часовою міткою, і до уваги беруться лише події в межах заданого часового вікна.

Нехай:

- $N_{F,1}$ — кількість унікальних користувачів, які досягли першого кроку воронки s_1;
- $N_{F,k}$ — кількість унікальних користувачів, які досягли кроку s_k ($k = 2,\ldots,K$);
- $N_{F,K}$ — кількість користувачів, які завершили всю воронку, тобто послідовно пройшли всі K кроків.

Тоді загальний коефіцієнт конверсії воронки визначається як:

$$CR_F = \frac{N_{F,K}}{N_{F,1}}. \tag{2.13}$$

Для аналізу «вузьких місць» додатково розглядаються локальні коефіцієнти переходу між кроками, які вже були введені у (2.6):

$$CR_{F,k\to k+1} = \frac{N_{F,k+1}}{N_{F,k}}, k = 1,\dots,K-1. \quad (2.14)$$

Ці показники відображають частку користувачів, що переходять з кроку s_k на крок s_{k+1}, і дозволяють виділити етапи з найбільшими втратами.

Крім кількісних показників переходу, у системі також оцінюються часові характеристики воронок. Для кожного кроку може обчислюватися середній час досягнення цього кроку від моменту входу у воронку, що дозволяє виявляти затримки у користувацьких сценаріях. Для цього на основі часових міток подій користувача визначається різниця між часом першого кроку та часом кожного наступного кроку, після чого ці значення агрегуються для всіх користувачів, що досягли відповідного кроку.

Аналіз воронок може виконуватися як у загальному вигляді, так і з урахуванням A/B експериментів. У цьому випадку події додатково фільтруються за *experiment_id* та *variant_id*, що дозволяє порівнювати:

- значення CR_F для різних варіантів одного експерименту;
- локальні коефіцієнти переходу $CR_{F,k\to k+1}$ між варіантами;
- зміни у структурі відтоку користувачів на окремих кроках воронки.

Таким чином, модель воронок конверсії інтегрує подієво-орієнтований підхід до збору даних із формальним описом кроків користувацького шляху та набором агрегованих показників, що дозволяють оцінювати як загальний результат, так і поведінку користувачів на кожному етапі. Це створює підґрунтя для прийняття рішень щодо оптимізації інтерфейсу та покращення сценаріїв взаємодії з веб-ресурсами клієнтів, у тому числі в контексті A/B тестування.

Подальші підрозділи другого розділу будуть присвячені аналізу існуючих рішень для зберігання й обробки подієвих даних та обґрунтуванню вибору

технологічного стеку й архітектурних підходів для реалізації зазначених моделей у межах інформаційно-аналітичної системи.

2.1.5. Моделювання статистичної значущості результатів

Для того щоб результати A/B тестування можна було інтерпретувати як такі, що мають практичний сенс, недостатньо порівняти лише значення конверсій або інших метрик. Необхідно оцінити, чи є спостережувана різниця між варіантами статистично значущою, тобто чи може вона бути пояснена випадковими коливаннями даних. У проєктованій системі реалізовано обчислення:

- коефіцієнтів конверсії для кожного варіанта;
- відносного покращення (uplift) тестових варіантів відносно контрольного;
- Z-тесту для двох пропорцій;
- Chi-Square тесту;
- довірчих інтервалів для конверсій.

Нехай для двох варіантів маємо кількість конверсій C_1,C_2 та кількість показів (користувачів) N_1,N_2.

Оцінки конверсій для кожного варіанта задаються як:

$$p_1 = \frac{C_1}{N_1}, p_2 = \frac{C_2}{N_2}. \tag{2.15}$$

Для перевірки гіпотези про рівність пропорцій p1p_1p1 та p2p_2p2 застосовується Z-тест із використанням «спільної» (пулінгової) оцінки пропорції:

$$p = \frac{C_1+C_2}{N_1+N_2}. \tag{2.16}$$

Тоді Z-статистика має вигляд:

$$z = \frac{p_1 - p_2}{\sqrt{p(1-p)\left(\frac{1}{N_1} + \frac{1}{N_2}\right)}}. \quad (2.17)$$

Отримане значення z порівнюється з критичними значеннями стандартного нормального розподілу, що дозволяє оцінити p-value та зробити висновок, чи є відмінність між варіантами статистично значущою на заданому рівні значущості.

Для одиничного варіанта з конверсією p та кількістю спостережень N довірчий інтервал для пропорції може бути заданий у вигляді:

$$CI = p \pm 1.96 \cdot \sqrt{\frac{p(1-p)}{N}}, \quad (2.18)$$

що відповідає 95% довірчому інтервалу в припущенні нормальної апроксимації біноміального розподілу. Довірчі інтервали для кожного варіанта створюють додаткову інтерпретаційну опору для аналізу: якщо інтервали суттєво перекриваються, різниця між варіантами може бути статистично незначущою.

Окрім Z-тесту, у системі також використовується Chi-Square тест для оцінювання узгодженості розподілу конверсій між варіантами із припущенням про відсутність ефекту. Він застосовується до контингентних таблиць «варіант × результат (конверсія/без конверсії)» і дозволяє перевіряти ті самі гіпотези в альтернативній формі.

Зазначені статистичні критерії інтегровані у модуль аналітики системи та використовуються для розрахунку показників у дашборді: відображення конверсій, відносного покращення, довірчих інтервалів і рівня статистичної значущості різниці між варіантами. Це забезпечує можливість ухвалювати рішення щодо впровадження або відхилення тестових варіантів не лише на основі візуальних відмінностей у метриках, а й із урахуванням статистичної надійності результатів.

У межах підрозділу 2.1 було формалізовано основні компоненти моделі оцінювання ефективності веб-ресурсів на основі A/B тестування:

- описано систему метрик, включаючи коефіцієнти конверсії, інтенсивність цільових дій, середній дохід на користувача та локальні коефіцієнти переходу між кроками воронок;
- визначено модель розподілу користувачів між варіантами експерименту на основі детермінованого хешування та принципів sticky bucketing;
- сформовано подієво-орієнтовану модель збору даних і структуру подій, що забезпечує гнучкий аналіз поведінки користувачів та прив'язку кожної події до експерименту й варіанта;
- побудовано формальну модель воронок конверсії, яка дозволяє аналізувати шляхи користувачів та виявляти «вузькі місця» в сценаріях взаємодії;
- визначено критерії статистичної значущості результатів (Z-тест, Chi-Square тест, довірчі інтервали), що дають можливість оцінювати надійність відмінностей між варіантами.

У сукупності ці моделі безпосередньо відображаються в реалізованій системі (SDK, API, сховище подій, дашборд) і формують теоретико-методичну основу для подальшого проєктування архітектури та вибору технологічного стеку, що розглядатиметься в наступних підрозділах розділу 2.

2.2. Огляд та аналіз існуючих рішень для A/B тестування та продуктової аналітики

У підрозділі 2.1 було формалізовано основні задачі, які має розв'язувати інформаційно-аналітична система: проведення A/B тестування, подієво-орієнтований збір даних, побудова воронок конверсії та оцінювання статистичної значущості результатів. На цьому етапі доцільно оцінити, наскільки існуючі класи рішень для A/B тестування та продуктової аналітики здатні підтримати ці задачі в умовах діяльності ФОП «Горенко Тарас Анатолійович».

Умовно можна виділити три основні групи рішень:

1) готові SaaS-платформи A/B тестування (корпоративні рішення);

2) платформи продуктової аналітики загального призначення;
3) власні рішення на базі подієвого сховища подій та кастомного інтерфейсу.

До першої групи належать корпоративні платформи A/B тестування, орієнтовані на середні та великі компанії. Вони, як правило, надають розвинений інструментарій для конфігурації експериментів, складних схем таргетингу й сегментації, персоналізації та роботи з великими обсягами трафіку. Такі системи добре підтримують повний цикл роботи з експериментами, але мають низку особливостей, критичних для невеликого підприємства, що працює з численними клієнтськими веб-ресурсами:

- високий поріг входу за вартістю підписки;
- берігання подієвих даних у хмарній інфраструктурі постачальника, що обмежує контроль над даними;
- необхідність адаптувати внутрішні процеси під модель роботи платформи;
- складність централізованого аналізу результатів для багатьох різних клієнтів у єдиній власній моделі даних.

До другої групи належать платформи продуктової аналітики (системи веб- та подієвої аналітики), основним призначенням яких є збір подієвих даних, побудова звітів, сегментація аудиторії, аналіз воронок і поведінки користувачів у часі. Частина таких систем підтримує базові механізми A/B тестування, однак експерименти розглядаються як додатковий модуль поверх загальної аналітики. Типовими особливостями цього класу рішень є:

- гнучкі можливості побудови воронок, сегментації та аналізу поведінки користувачів;
- власна схема зберігання подій і модель ідентифікації користувачів;
- обмежена можливість повноцінно керувати життєвим циклом A/B експериментів у розрізі багатьох клієнтських проєктів;
- труднощі інтеграції з уже наявною архітектурою (клієнтський SDK, бекенд, сховище подій), якщо підприємство прагне мати єдине джерело даних.

Третю групу становлять власні рішення на базі подієвого сховища подій та кастомного інтерфейсу, коли підприємство само будує інфраструктуру збору та

обробки подій (наприклад, використовуючи аналітичну СУБД стовпчикового типу для зберігання подій, бекенд для приймання та попередньої обробки даних, клієнтський SDK для відстеження подій і призначення варіантів, а також окремий інтерфейс для побудови воронок та аналізу результатів експериментів). У такому підході:

- схема подій, ідентифікація користувачів та сесій, структура воронок і цільових метрик повністю контролюються підприємством;
- забезпечується спільне сховище для всіх клієнтських веб-ресурсів, що дозволяє застосовувати єдині підходи до аналізу;
- можливе безпосереднє повторне використання вже наявних компонентів (у тому числі серверної інфраструктури, SDK, інтеграції з клієнтськими сайтами);
- вартість масштабування визначається обраними технологіями з відкритим вихідним кодом і власною архітектурою, а не тарифами SaaS-постачальників.

Для систематизації відмінностей між цими підходами доцільно розглянути їх за рядом критеріїв, релевантних задачам, сформульованим у підрозділі 2.1. Узагальнені результати подано в таблиці 2.2.

Таблиця 2.2 – Порівняння класів рішень для A/B тестування та продуктової аналітики

Критерій	**Корпоративні SaaS-платформи A/B тестування**	**Платформи продуктової аналітики**	**Власна подієво-орієнтована система**
Контроль над даними	Дані зберігаються у постачальника	Дані зберігаються у постачальника	Дані зберігаються у власній інфраструктурі
Гнучкість схеми подій	Обмежена рамками платформи	Гнучка, але в межах моделі постачальника	Повністю визначається підприємством

Підтримка повного циклу A/B експериментів	Так, із розширеними можливостями	Частково, A/B як додатковий модуль	Реалізується відповідно до власних моделей

Закінчення таблиці 2.2

Критерій	**Корпоративні SaaS-платформи A/B тестування**	**Платформи продуктової аналітики**	**Власна подієво-орієнтована система**
Підтримка воронок конверсії	Є, але переважно в контексті експериментів	Є, як частина продуктової аналітики	Гнучка, з урахуванням специфіки клієнтів
Адаптація під кілька клієнтських веб-ресурсів	Орієнтація на окремий продукт/сервіс	Орієнтація на продукт чи набір продуктів	Орієнтація на портфель клієнтських проєктів
Інтеграція з наявним SDK та бекендом	Обмежена; потрібна адаптація	Обмежена; потрібна адаптація	Повна, оскільки SDK і бекенд розробляються спільно
Вартість та поріг входу	Високий для невеликих студій	Залежить від тарифів, може бути суттєвим	Контролюється вибором технологій

Як видно з таблиці 2.2, корпоративні SaaS-платформи A/B тестування забезпечують високий функціональний рівень, але їх застосування у контексті ФОП «Горенко Тарас Анатолійович» є економічно та організаційно недоцільним. Платформи продуктової аналітики надають потужні інструменти для аналізу поведінки користувачів і воронок, але не дозволяють органічно інтегрувати A/B

тестування у власну архітектуру збору подієвих даних і роботу з багатьма клієнтськими веб-ресурсами одночасно.

Натомість підхід, що базується на побудові власної подієво-орієнтованої системи з використанням окремого сховища подій, серверних компонентів та клієнтського SDK, найкраще відповідає задачам, сформульованим у попередніх підрозділах. Він дозволяє:

- реалізувати моделі, описані в 2.1 (A/B тестування, воронки, статистичний аналіз) в єдиній архітектурі;
- забезпечити централізований збір та аналіз подій для всіх клієнтських веб-ресурсів підприємства;
- адаптувати інтерфейс аналітики та стек технологій до потреб конкретного бізнес-середовища.

Таким чином, результати огляду та аналізу існуючих рішень підтверджують доцільність обраного в роботі напряму — розробки власної інформаційно-аналітичної системи A/B тестування та продуктової аналітики, що базується на подієво-орієнтованому сховищі даних та інтегрованому клієнтському SDK. У наступних підрозділах розділу 2 буде обґрунтовано вибір конкретного технологічного стеку та архітектурних рішень для реалізації цієї системи.

2.3. Обґрунтування вибору технологій, методів та алгоритмів

2.3.1. Вибір технологічного стеку

Для реалізації інформаційно-аналітичної системи A/B тестування та аналізу воронок конверсії обрано технологічний стек, який поєднує засоби для побудови веб-інтерфейсу, зберігання метаданих експериментів та високопродуктивної обробки подієвих даних.

Next.js 14 (App Router, TypeScript) використовується як основа для фронтенду та бекенду системи. Такий вибір обґрунтовується тим, що:

а) забезпечується єдиний стек для розробки інтерфейсу дашборду та серверних API-ендпоінтів (зокрема, /api/track, /api/bucket, /api/experiments), що спрощує підтримку та розгортання системи;

b) підтримка серверного рендерингу (SSR) та статичної генерації (SSG) дозволяє будувати швидкі та відгукливі сторінки аналітичної панелі;
c) використання TypeScript підвищує надійність коду за рахунок статичної типізації та полегшує рефакторинг системи в процесі розвитку;
d) вбудована модель маршрутизації App Router спрощує організацію модулів дашборду (сторінок проєктів, експериментів, воронок, налаштувань тощо).

Supabase (PostgreSQL) виконує роль основного сховища метаданих експериментів. У цій базі даних зберігаються проєкти, експерименти, варіанти, цілі, визначення воронок, таблиця призначень користувачів (user_buckets) та інші конфігураційні сутності. Обґрунтування вибору Supabase полягає в тому, що:

a) використовується знайомий реляційний SQL-двигун PostgreSQL, який добре підходить для зберігання структурованих конфігураційних даних;
b) платформа надає вбудовані механізми автентифікації та політик на рівні рядків (RLS), що спрощує реалізацію розмежування доступу;
c) доступна зручна адміністративна панель для керування схемою даних і контентом без додаткових інструментів;
d) CRUD-операції з метаданими експериментів легко інтегруються з Next.js API.

Для зберігання та аналітики сирих подієвих даних використовується ClickHouse як стовпчикова аналітична СУБД. Даний вибір пов’язаний із такими особливостями:

a) ClickHouse оптимізований для обробки великих обсягів подієвих даних з високою швидкістю агрегації, що є критичним для задач аналізу конверсій та воронок;
b) стовпчикова організація зберігання та можливість партиціювання дозволяють ефективно виконувати запити з фільтрацією за часом, експериментом, варіантом та іншими атрибутами;
c) підтримка матеріалізованих подань дає змогу будувати похідні таблиці з агрегованими метриками (наприклад, погодинна чи денна статистика експериментів), що прискорює відображення даних у дашборді;

d) інтеграція зі схемою подій системи (таблиця events) дозволяє прямо реалізувати подієво-орієнтовану модель, описану раніше.

Для побудови інтерфейсу дашборду використовуються:

a) Tailwind CSS та компоненти ShadCN UI — для створення уніфікованого, адаптивного, компонентно-орієнтованого інтерфейсу без надмірної верстки «з нуля»;
b) Recharts — для візуалізації даних A/B експериментів, воронок конверсії та часових рядів метрик у вигляді графіків та діаграм;
c) Docker — для контейнеризації ClickHouse і, за потреби, інших компонентів, що спрощує розгортання та повторюваність середовищ.

У сукупності цей технологічний стек забезпечує:

a) узгоджену архітектуру клієнтського SDK, серверних API та аналітичного дашборду;
b) розділення відповідальності між сховищем метаданих (Supabase) та аналітичним сховищем подій (ClickHouse);
c) достатню масштабованість для роботи з декількома клієнтськими веб-ресурсами та великою кількістю подій.

2.3.2. Обґрунтування вибору алгоритмів та методів

Вибір алгоритмів та методів у системі ґрунтується на задачах, сформульованих у підрозділі 2.1, і особливостях архітектури, описаної в документації.

Для розподілу користувачів між варіантами експериментів використовується підхід, заснований на детермінованому хешуванні (consistent hashing) у поєднанні з принципом sticky bucketing. Такий вибір обґрунтовується тим, що:

- хеш-функція над парою user_id та experiment_id дозволяє реалізувати розподіл користувачів без необхідності зберігати стан на сервері в найпростішому варіанті;

- за рахунок нормування хеш-значення та розбиття інтервалу [0,1)[0,1)[0,1) відповідно до ваг варіантів досягається наближення фактичних часток трафіку до заданих;
- sticky bucketing (запис призначеного варіанта в таблицю user_buckets і кешування на стороні клієнта) гарантує, що користувач під час повторних відвідувань отримує той самий варіант, що підвищує коректність статистичного аналізу й стабільність користувацького досвіду.

Для зберігання та аналізу взаємодій користувачів обрана подієво-орієнтована модель даних, у якій кожна дія користувача представляється окремою подією з часовою міткою. Ця модель:

- забезпечує гнучкість при побудові різних типів аналітики, зокрема розрахунку конверсій, аналізу воронок та кастомних метрик;
- добре узгоджується з можливостями ClickHouse як аналітичної СУБД, де подієві дані можуть ефективно агрегуватися та фільтруватися;
- дозволяє легко розширювати схему подій за рахунок додаткових властивостей (properties) без зміни базової структури.

Для оцінювання результатів A/B експериментів використовуються класичні статистичні методи:

- Z-тест для двох пропорцій — для перевірки гіпотез щодо відмінностей конверсій між варіантами;
- Chi-Square тест — як альтернативний критерій для аналізу контингентних таблиць «варіант × результат»;
- довірчі інтервали для пропорцій — для інтерпретації оцінок конверсій із врахуванням статистичної невизначеності.

Ці методи є стандартними для задач аналізу експериментів, легко інтерпретуються (через p-value та рівні значущості) та вже реалізовані в серверній частині системи на основі формул, закладених у ресурсних матеріалах.

На рівні клієнтського SDK реалізовано пакетування подій та повторні спроби відправлення (retry logic). Події групуються в пакети обмеженого розміру

або за часом, після чого надсилаються на бекенд у запиті до /api/track. Такий підхід:

- зменшує кількість HTTP-запитів із клієнтського сайту;
- знижує вплив механізму відстеження на продуктивність веб-ресурсів клієнтів;
- дозволяє коректно обробляти тимчасові збої мережі завдяки повторним спробам надсилання, що узгоджується з відповідною BPMN-діаграмою обробки помилок.

2.4. Висновки до розділу 2

Використання Next.js 14, Supabase (PostgreSQL) та ClickHouse забезпечує узгоджену архітектуру для дашборду, API та сховищ даних, а також необхідну продуктивність і масштабованість. Обрані алгоритми — детерміноване хешування, sticky bucketing, подієво-орієнтована модель даних та класичні статистичні методи — безпосередньо відповідають формалізованим у підрозділі 2.1 задачам системи та підтримуються структурою даних і потоками, описаними в документації.

Отримані результати створюють основу для подальшого переходу до проєктування архітектури системи та її програмної реалізації, що розглядатиметься у наступному розділі.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА АПРОБАЦІЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ

3

3.1. Проєктування та створення аналітичної системи

На основі сформульованих у другому розділі моделей A/B тестування, подієво-орієнтованого збору даних, аналізу воронок конверсії та статистичної оцінки результатів розроблено інформаційно-аналітичну систему, призначену для роботи з клієнтськими веб-ресурсами підприємства. У цьому підрозділі розглядається загальна структура системи, склад її основних компонентів та логіка організації дашборду для роботи з експериментами та аналітикою.

3.1.1. Загальна структура системи

Інформаційно-аналітична система має багаторівневу архітектуру та включає такі основні компоненти: клієнтський рівень, SDK, серверну частину (API та дашборд) і рівень даних. Узагальнену архітектуру подано на рисунку 3.1.

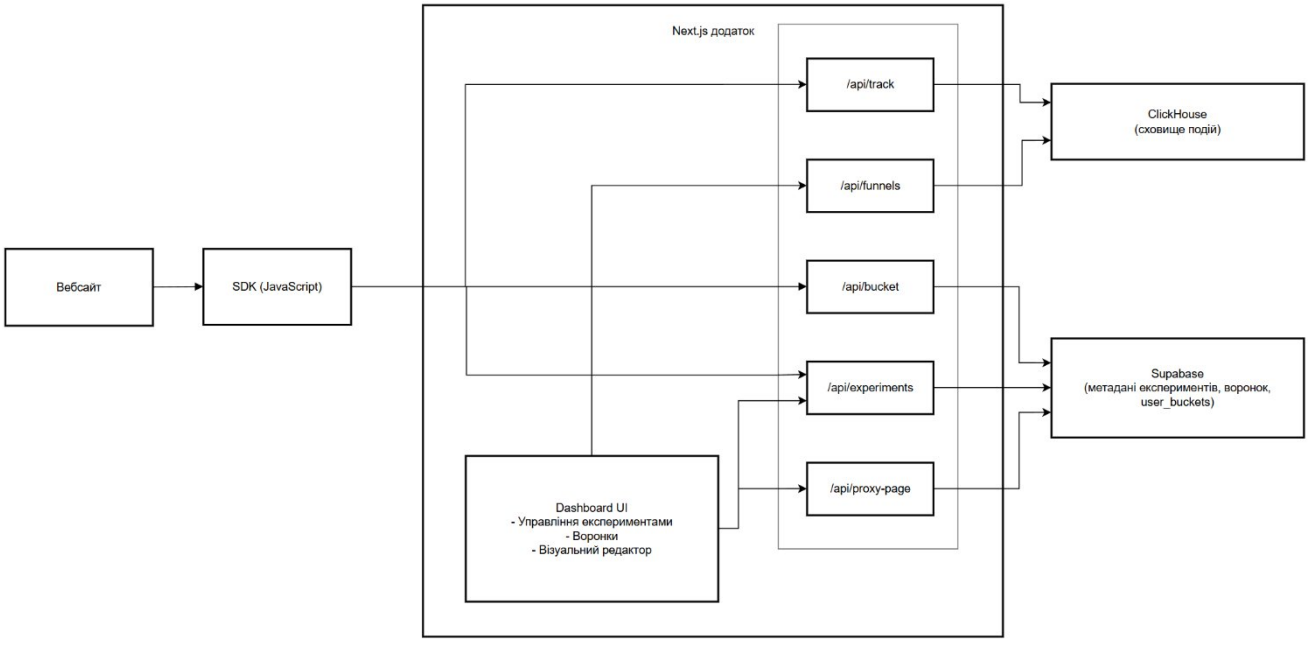


Рисунок 3.1 — Загальна архітектура інформаційно-аналітичної системи A/B тестування

Клієнтський рівень охоплює веб-ресурси клієнтів (інтернет-магазини, лендінги, сервісні сайти), до яких підключається клієнтський SDK. Окремо передбачено демо-сайт, що імітує типовий e-commerce-флоу з кількома основними сторінками (головна, сторінка товару, кошик, оформлення замовлення). Демо-сайт використовується для тестування, налагодження та демонстрації роботи системи без втручання у реальні проєкти клієнтів.

SDK реалізовано як клієнтську JavaScript-бібліотеку, що інтегрується у код веб-ресурсу. Він забезпечує:

- ініціалізацію відстеження (init), включаючи генерацію або зчитування стійкого user_id і session_id;
- отримання призначених варіантів для активних експериментів (getVariant, getFeatureFlag) через звернення до серверного API;
- реєстрацію подій користувацької активності (track, спеціалізовані обгортки на кшталт trackClick, trackConversion);
- пакетування подій у батчі та повторні спроби відправлення у випадку тимчасових помилок мережі;
- застосування візуальних змін на стороні клієнта відповідно до конфігурації варіанту (наприклад, зміна текстів кнопок, елементів інтерфейсу).

Серверна частина реалізована на основі Next.js (App Router) і поєднує функціональність API та дашборду. До складу API входять:

- POST /api/track — приймання та обробка пакетів подій від SDK, валідація та запис у сховище подій;
- POST /api/bucket — призначення варіантів користувачам на основі детермінованого розподілу (хешування, ваги варіантів, sticky bucketing);
- POST /api/experiments/active — отримання списку активних експериментів для поточного URL, що використовується SDK при ініціалізації;
- GET/POST/PATCH/DELETE /api/experiments/* — операції створення, читання, оновлення та видалення записів про експерименти;
- GET/POST /api/funnels/* — керування визначеннями воронок та отримання результатів їх аналізу;

- /api/proxy-page — проксіювання сторінок клієнтського сайту для роботи візуального редактора, що дозволяє застосовувати варіанти інтерфейсу в середовищі попереднього перегляду.

Рівень даних поділено на два логічні контури. Перший — це Supabase (PostgreSQL), що використовується для зберігання метаданих і конфігурацій:

- проєкти та пов’язані з ними веб-ресурси;
- експерименти, їх варіанти та цілі;
- визначення воронок та кроків у воронках;
- таблиця призначень користувачів (user_buckets), що забезпечує sticky bucketing;
- інші службові сутності, пов’язані з налаштуванням системи.

Другий контур — ClickHouse як аналітичне сховище подієвих даних. У ньому зберігаються:

- таблиця events із сирими подіями, що надходять із SDK;
- матеріалізовані уявлення для агрегованої статистики експериментів (погодинні та денні показники);
- допоміжні уявлення для побудови воронок та розрахунку пов’язаних метрик.

Такий розподіл забезпечує чітке розмежування між конфігураційними метаданими та високочастотним потоком подій, що відповідає вимогам до продуктивності й масштабованості системи.

3.1.2. Компоненти дашборду

Дашборд є основним інтерфейсом взаємодії користувача з системою з боку підприємства. Він реалізований як набір сторінок та компонентів у Next.js і забезпечує повний цикл роботи з проєктами, експериментами та аналітикою.

До основних компонентів дашборду належать:

- Список експериментів — сторінка, на якій у табличному вигляді відображаються всі експерименти для обраного проєкту. Передбачено базові можливості фільтрації (за статусом, типом, датою створення),

сортування та пошуку. Для кожного експерименту вказуються ключові атрибути: назва, статус (чернетка, активний, призупинений, завершений), дати початку та завершення.

- Майстер створення експерименту — послідовність кроків (приблизно п’ять екранів), що проводять користувача через процес налаштування експерименту. На різних кроках задаються базова інформація (назва, опис, гіпотеза), варіанти та їх ваги, цілі (метрики), умови старту та завершення експерименту. Майстер забезпечує перевірку коректності введених даних перед збереженням конфігурації.
- Сторінка деталей експерименту — інтерфейс для перегляду та аналізу конкретного експерименту. На цій сторінці відображаються основні відомості про експеримент, список варіантів, поточні значення ключових метрик (конверсій, доходу, показників воронок), а також результати статистичної оцінки (різниця між варіантами, довірчі інтервали, p-value). Передбачено можливість змінювати статус експерименту (активація, призупинення, завершення) та за потреби коригувати окремі параметри.
- Модуль побудови воронок — окремий інтерфейс, у якому користувач може визначати воронки на основі подій, що надходять у систему. У модулі задається послідовність кроків воронки через вибір event_name, умов на URL та додаткових параметрів. Після побудови воронки дашборд відображає кількість користувачів на кожному кроці, коефіцієнти переходу та загальний коефіцієнт конверсії воронки, а також дозволяє аналізувати ці показники окремо за варіантами експериментів.
- Візуальний редактор — компонент, що працює на основі вбудованого iframe із проксійованою сторінкою клієнтського сайту. У цьому редакторі користувач може обирати елементи інтерфейсу, змінювати їхні властивості (текст, стилі, атрибути) та прив’язувати ці зміни до конкретних варіантів експерименту. Редактор взаємодіє з API /api/proxy-page та конфігурацією експериментів, забезпечуючи попередній перегляд варіантів без прямого втручання в код сайту.

- Сторінки налаштування проєктів і API-ключів — інтерфейси для створення та керування проєктами (веб-ресурсами клієнтів), генерації та перегляду API-ключів, що використовуються для інтеграції SDK. Тут же можуть задаватися базові параметри проєкту (домен, тип сайту, часовий пояс тощо).

Структура дашборду побудована таким чином, щоб забезпечити логічний перехід від загального огляду портфеля експериментів до детального аналізу конкретних варіантів і воронок, а також підтримати повний життєвий цикл експерименту — від створення гіпотези до ухвалення рішень на основі отриманих даних.

Отже, система побудована як сукупність взаємодіючих модулів: клієнтський SDK, серверне API на базі Next.js, дашборд для роботи з експериментами та двокомпонентний рівень даних, що включає Supabase для метаданих і ClickHouse для подієвих даних.

Обрана архітектура забезпечує модульність і розширюваність: кожен компонент може розвиватися незалежно, при цьому зберігається узгодженість загальної моделі даних та інтерфейсів взаємодії. Такий підхід дозволяє адаптувати систему до зростання кількості клієнтських веб-ресурсів, обсягу подієвих даних та складності аналітичних задач.

Отримані результати створюють основу для подальшого аналізу джерел даних і можливостей апробації системи на реальних і тестових веб-ресурсах, що розглядатиметься у наступних підрозділах розділу 3.

3.2. Дослідження та аналіз джерел даних для перевірки обраних технологій

3.2.1. Джерела даних

У реалізованій інформаційно-аналітичній системі основним джерелом даних є події, що збираються клієнтським SDK з веб-ресурсів, до яких він підключений. SDK інтегрується у код клієнтських сайтів і реєструє події, пов’язані з переглядом сторінок, взаємодією з елементами інтерфейсу та досягненням цільових дій. Саме на основі цих подій формуються показники

ефективності, розраховуються конверсії, будуються воронки та виконується аналіз A/B експериментів.

Для перевірки коректності роботи всієї аналітичної вертикалі — від збору подій на клієнті до візуалізації результатів у дашборді — у системі використовується спеціально підготовлений демо-сайт. Він імітує типовий e-commerce-сценарій і містить послідовність базових сторінок:

- головна сторінка;
- сторінка товару;
- сторінка кошика;
- сторінка оформлення замовлення.

На кожній із цих сторінок у коді передбачено виклики методів SDK, які генерують події, що відповідають основним крокам користувацького шляху: перегляд сторінки, перехід до товару, додавання до кошика, оформлення замовлення тощо. Таким чином, демо-сайт забезпечує відтворювану послідовність дій користувача, на основі якої можна:

- перевірити коректність відстеження подій та їх передачі до серверної частини системи;
- протестувати побудову воронок конверсії за подіями та URL;
- оцінити поведінку системи при обробці потоку подій у режимі наближеному до реального сценарію роботи з e-commerce-ресурсом.

У реальних умовах роботи системи аналогічний підхід використовується для клієнтських веб-ресурсів підприємства: SDK підключається до сторінок сайтів, де відстежуються події, пов’язані з бізнес-процесами конкретного замовника. Однак у межах даної роботи ключова увага зосереджена на даних, отриманих із демо-сайту, оскільки вони дозволяють репродукувати однакові сценарії для порівняння варіантів і перевірки працездатності обраних технологій і моделей.

3.2.2. Структура подій як даних для аналізу

Події, що надходять від клієнтського SDK і зберігаються в ClickHouse, мають уніфіковану структуру, орієнтовану на подальший аналіз A/B експериментів та воронок конверсії. Для кожної події фіксуються такі групи атрибутів:

- ідентифікатори користувача та сесії — дозволяють об’єднувати окремі події в послідовності дій одного користувача та аналізувати поведінку в межах сесій;
- ідентифікатори експерименту та варіанту — вказують, у межах якого A/B експерименту та якого варіанта було зафіксовано подію, що забезпечує можливість розрахунку метрик для кожного варіанта окремо;
- тип події (наприклад, impression, click, conversion, custom) — описує загальну природу дії користувача;
- назва події (event_name) — деталізує конкретну дію (перегляд певної сторінки, клік по визначеному елементу, завершення замовлення тощо);
- URL сторінки — дозволяє прив’язати подію до конкретного ресурсу або типу сторінки на сайті;
- довільні властивості (properties) — слугують для передавання додаткових параметрів події (наприклад, суми замовлення, ідентифікатора товару та інших бізнес-атрибутів);
- часова мітка та інформація про пристрій — фіксують момент настання події та характеристики середовища (тип пристрою, браузер, операційна система тощо).

Така структура є цілеспрямовано мінімальною, але достатньою для основних задач системи:

- вона забезпечує розрахунок показників конверсії та пов’язаних метрик для A/B експериментів;
- дозволяє будувати воронки конверсії за послідовностями подій і шаблонами URL;

- не містить надлишкових атрибутів, які не використовуються в аналізі, що зменшує обсяг зберігання та спрощує схему даних.

Завдяки уніфікованій структурі подій стає можливим виконання однакових аналітичних сценаріїв як для демо-сайту, так і для різних клієнтських веб-ресурсів, що особливо важливо для підприємства, яке працює з портфелем проєктів.

Структура подій, що зберігаються в ClickHouse, побудована навколо ідентифікаторів користувача, сесії, експерименту та варіанту, типу й назви події, URL та додаткових властивостей. Вона повністю відповідає вимогам, сформульованим у другому розділі, та можливостям обраної аналітичної СУБД. Такий підхід забезпечує єдину подієву модель, яка придатна для розрахунку конверсій, побудови воронок і оцінювання результатів A/B експериментів як для тестового демо-сайту, так і для реальних клієнтських веб-ресурсів.

Отримані результати створюють підґрунтя для подальшого проєктування та реалізації сховища даних, що поєднує конфігураційну інформацію про експерименти та подієві дані, — це є предметом підрозділу 3.3.

3.3. Проєктування та створення сховища даних

Сховище даних інформаційно-аналітичної системи A/B тестування побудовано за двоконтурною схемою:

1. реляційний контур на базі Supabase (PostgreSQL) для зберігання метаданих про проєкти, експерименти, варіанти, цілі та воронки;
2. аналітичний контур на базі ClickHouse для зберігання подієвих даних, що надходять від клієнтського SDK і використовуються для розрахунку метрик, побудови воронок та оцінювання результатів експериментів.

Такий підхід дозволяє відокремити конфігураційну інформацію, яка змінюється відносно рідко й потребує цілісності, від високочастотного потоку подій, що вимагає спеціалізованих засобів аналітичної обробки.

3.3.1. Логічна модель сховища даних

У реляційному сховищі на базі Supabase зберігаються метадані, необхідні для опису структури A/B експериментів, варіантів, цілей, воронок та доступу до системи. Основні таблиці та їх призначення наведено в таблиці 3.1.

Таблиця 3.1 – Основні таблиці бази даних Supabase та їх призначення

Таблиця	**Призначення**
projects	Зберігає проєкти (робочі простори), що відповідають окремим клієнтським веб-ресурсам або їх групам
experiments	Містить конфігурацію A/B експериментів (назва, опис, статус, тип, часові параметри)
variants	Описує варіанти експериментів із заданими вагами трафіку та конфігурацією інтерфейсних змін
goals	Містить цілі та ключові метрики, пов’язані з конкретними експериментами
api_keys	Зберігає API-ключі, що використовуються для автентифікації SDK та зовнішніх інтеграцій
funnels	Містить визначення воронок конверсії для проєктів
funnel_steps	Описує послідовні кроки воронок із зазначенням подій і умов їх відбору
user_buckets	Зберігає призначення «користувач–варіант експерименту» (реалізація механізму sticky bucketing)
events	Зберігає метадані подій (за потреби службових задач; основний масив подій зберігається в ClickHouse)

Логічна схема бази даних Supabase з основними зв’язками між таблицями наведена на рисунку 3.2.

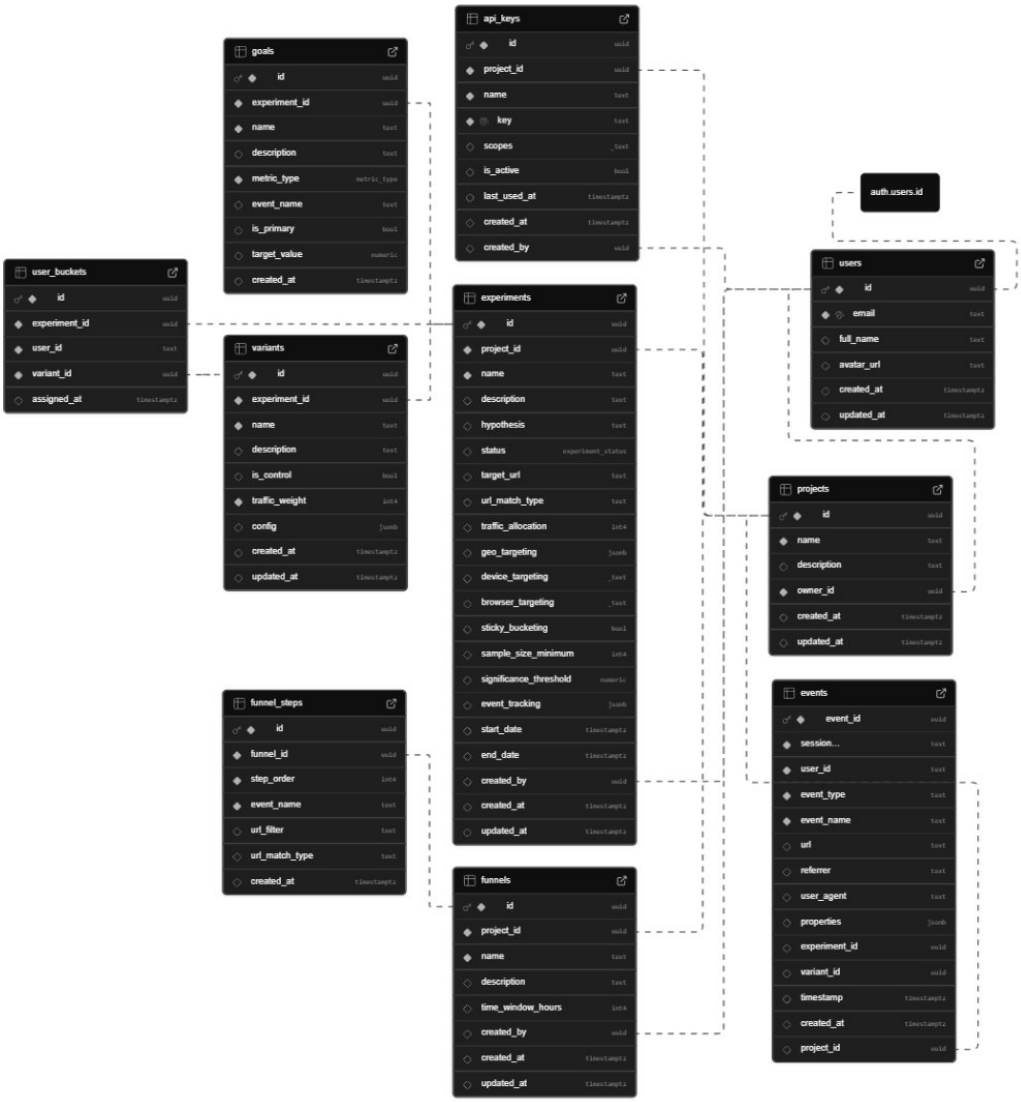


Рисунок 3.2 – Логічна схема бази даних Supabase для зберігання метаданих інформаційно-аналітичної системи A/B тестування

Основні зв’язки між таблицями Supabase згруповано в таблиці 3.2. Вона відображає, як конфігураційні сутності пов’язані між собою в єдиній моделі даних.

Таблиця 3.2 – Ключові зв’язки між таблицями бази даних Supabase

Батьківська таблиця	**Дочірня таблиця**	**Тип зв’язку**	**Опис зв’язку**
projects	experiments	Один-до-багатьох	Кожен проєкт може містити кілька A/B експериментів

Закінчення таблиці 3.2

Батьківська таблиця	**Дочірня таблиця**	**Тип зв’язку**	**Опис зв’язку**
experiments	variants	Один-до-багатьох	Для кожного експерименту задається один контрольний і один або кілька тестових варіантів
experiments	goals	Один-до-багатьох	До експерименту може бути прив’язано кілька цілей (метрик)
projects	funnels	Один-до-багатьох	Воронки конверсії визначаються окремо для кожного проєкту
funnels	funnel_steps	Один-до-багатьох	Кожна воронка складається з послідовності кроків з відповідними подіями та умовами відбору
experiments	user_buckets	Один-до-багатьох	Для кожного експерименту зберігається призначення варіантів конкретним користувачам
projects	api_keys	Один-до-багатьох	Для кожного проєкту зберігаються один або кілька API-ключів для інтеграції SDK та бекенд-сервісів

Такий набір сутностей і зв’язків забезпечує:

- цілісність конфігураційних даних: кожен експеримент, варіант, ціль або воронка завжди належать до конкретного проєкту;
- узгодженість між A/B тестуванням і аналізом воронок: визначення воронок у таблицях funnels і funnel_steps може безпосередньо використовуватися під час аналізу подій, пов’язаних з експериментами;

- коректну реалізацію механізму sticky bucketing: таблиця user_buckets зберігає стійкі призначення варіантів для користувачів, що важливо для інтерпретації подій у розрізі варіантів.

Повний SQL-опис структури таблиць Supabase (перелік полів, типів даних, первинних і зовнішніх ключів) наведено в додатку Б.

Аналітичне сховище ClickHouse використовується для зберігання подієвих даних і підготовки агрегованих представлень для дашборду. Основні об’єкти цього контуру наведено в таблиці 3.3.

Таблиця 3.3 – Основні таблиці та матеріалізовані подання в ClickHouse

Об’єкт	**Тип**	**Призначення**
events	таблиця	Зберігає сирі події, що надходять від SDK (перегляди, кліки, конверсії та інші дії користувачів)
variant_stats_hourly	materialized view	Формує погодинні агрегати за експериментами та варіантами (покази, конверсії, базові метрики)
experiment_stats_daily	materialized view	Формує денну статистику експериментів для оглядових звітів
funnel_step_events	materialized view	Відбирає події, що відповідають крокам визначених воронок, з урахуванням умов на події та URL
funnel_conversions	materialized view	Формує записи про завершені проходження воронок для розрахунку загальних і локальних конверсій

Таблиця events реалізує подієву модель, описану раніше: кожен запис відповідає одній дії користувача й містить ідентифікатори користувача, сесії, експерименту та варіанта, тип і назву події, URL, часову мітку та додаткові властивості. Матеріалізовані подання будуються поверх цієї таблиці та

забезпечують швидке отримання агрегованої статистики для дашборду без необхідності виконання ресурсоємних запитів по сирих даних.

Узгоджена логічна модель двох контурів сховища даних дає змогу використовувати Supabase для зберігання конфігураційних сутностей, а ClickHouse — для зберігання та аналітичної обробки подієвих даних, прив’язаних до цих конфігурацій.

3.3.2. Організація завантаження та обробки даних (ETL/ELT-процес)

У розробленій системі застосовується підхід, близький до ELT (Extract–Load–Transform): події спочатку завантажуються до аналітичного сховища, а подальші агрегування та підготовка представлень виконуються на рівні ClickHouse. Окремий офлайн ETL-конвеєр із пакетною обробкою даних не використовується; натомість реалізовано on-line потік обробки подій.

Загальний процес ініціалізації SDK, розподілу користувачів між варіантами та подальшої обробки подій і запису їх у сховище даних подано на діаграмі послідовності взаємодії компонентів (рисунок 3.3).

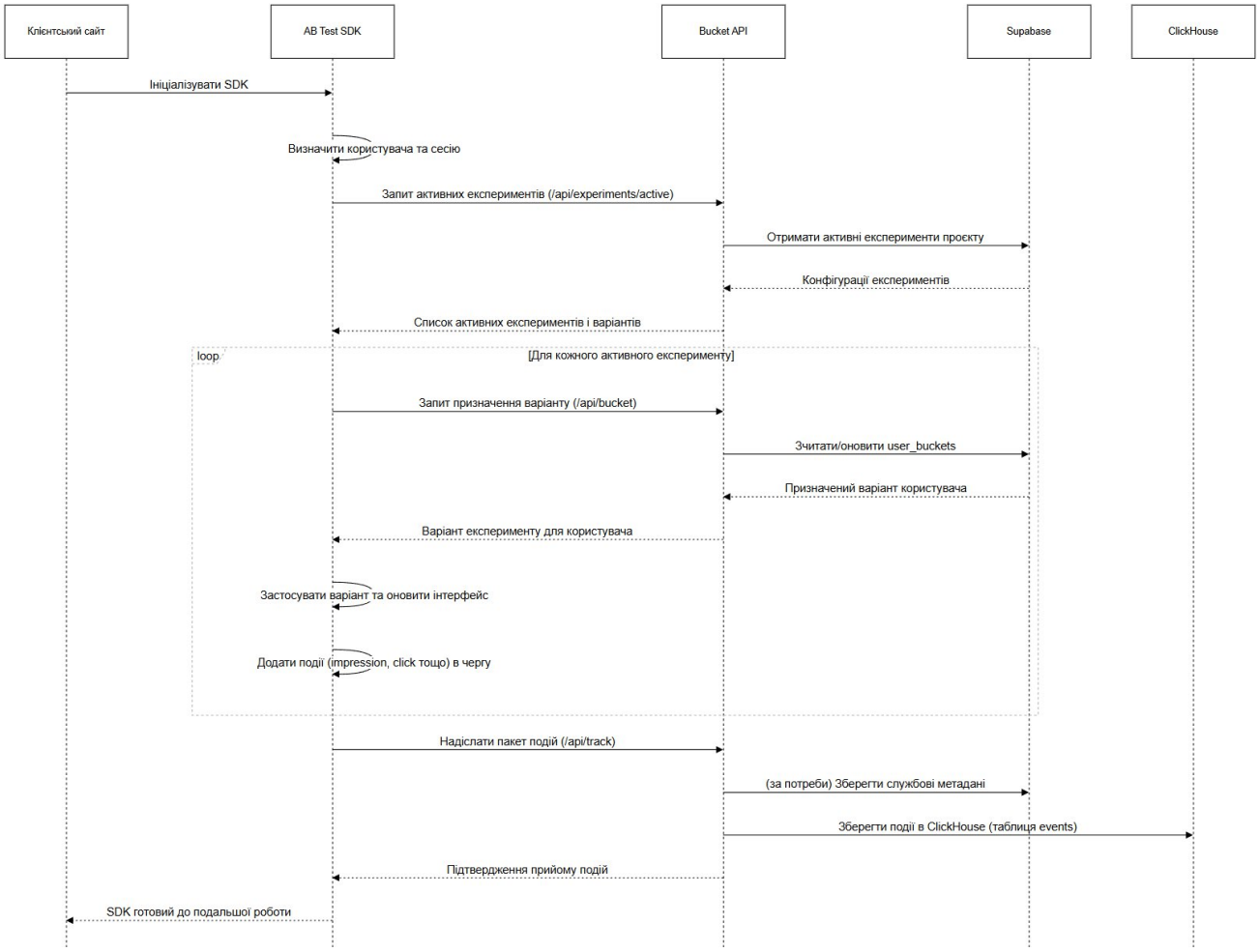


Рисунок 3.3 – Діаграма послідовності взаємодії компонентів

Послідовність кроків включає такі етапи:

1. Фіксація події на клієнтському сайті. На клієнтському веб-ресурсі, інтегрованому з SDK, під час взаємодії користувача з інтерфейсом (перегляд сторінки, перехід до товару, клік по кнопці, оформлення замовлення тощо) генеруються події. SDK формує події відповідно до єдиної структури (ідентифікатори користувача, сесії, експерименту, варіанта, тип і назва події, URL, властивості, часова мітка) та додає їх до локальної черги.
2. Формування та відправлення пакету подій. Події надсилаються на сервер порціями: SDK формує батч і виконує запит до серверного ендпоінта POST /api/track. Це дозволяє зменшити кількість HTTP-запитів і знизити

навантаження на клієнтський сайт. У запиті передаються також службові параметри, зокрема API-ключ проєкту.

3. Приймання та валідація подій на сервері. На стороні Next.js API ендпоінт POST /api/track:
    - перевіряє коректність API-ключа та визначає, до якого проєкту належать події;
    - виконує базову валідацію схеми подій (наявність обов'язкових полів, допустимі значення типів подій);
    - за потреби доповнює події інформацією про середовище (наприклад, дані з user-agent).

У разі виявлення помилок валідності некоректні події відкидаються або логуються окремо, що запобігає потраплянню неконсистентних даних до аналітичного сховища.

4. Запис подій до сховища. Після успішної валідації події записуються до таблиці events в ClickHouse. На цьому етапі дані зберігаються без суттєвих трансформацій, що відповідає підходу «спочатку load, потім transform». Таблиця організована таким чином, щоб забезпечити ефективну фільтрацію та агрегацію за часом, експериментом, варіантом та іншими ключовими атрибутами.
5. Формування агрегованих представлень. Вставка нових подій у таблицю events автоматично приводить до оновлення матеріалізованих подань:
    - variant_stats_hourly акумулює дані про кількість показів і конверсій у погодинному розрізі для кожного експерименту та варіанта;
    - experiment_stats_daily формує денні агрегати, які використовуються для оглядових звітів і відстеження динаміки показників;
    - funnel_step_events відбирає події, що відповідають крокам воронок, визначеним у таблицях funnels і funnel_steps;
    - funnel_conversions на основі послідовності подій фіксує завершені проходження воронок, необхідні для розрахунку коефіцієнтів конверсії на кожному кроці та загального коефіцієнта воронки.

Таким чином, значна частина перетворень даних виконується на рівні ClickHouse і не потребує окремих зовнішніх ETL-джобів.

6. Використання даних у дашборді**.** Дашборд, реалізований у Next.js, отримує конфігураційну інформацію про проєкти, експерименти, варіанти, цілі та воронки з Supabase, а фактичні значення метрик — із агрегованих представлень у ClickHouse. Це дозволяє:
    - відображати показники конверсій і пов’язані метрики для кожного варіанта експерименту;
    - аналізувати проходження воронок у розрізі експериментів і варіантів;
    - оцінювати ефективність змін на веб-ресурсах клієнтів на основі актуальних даних.

У сукупності описаний потік реалізує безперервне завантаження та перетворення даних, що забезпечує актуальність аналітики без складної зовнішньої ETL-інфраструктури.

Отже, у підрозділі 3.3 розглянуто логічну модель та організацію сховища даних інформаційно-аналітичної системи A/B тестування. Показано, що:

- використання Supabase (PostgreSQL) як реляційного сховища метаданих дає змогу структуровано зберігати інформацію про проєкти, експерименти, варіанти, цілі, воронки та призначення користувачів, забезпечуючи цілісність і узгодженість конфігураційних даних;
- застосування ClickHouse як аналітичного сховища подієвих даних дозволяє ефективно обробляти великі обсяги подій, виконувати агрегування та підтримувати побудову воронок і розрахунок метрик для A/B експериментів;
- реалізований on-line ELT-процес забезпечує завантаження подій у сховище в режимі, наближеному до реального часу, із подальшим формуванням агрегованих представлень засобами аналітичної СУБД без окремих офлайн ETL-конвеєрів.

Обрана архітектура сховища даних відповідає вимогам до системи, орієнтованої на роботу з декількома клієнтськими веб-ресурсами та інтенсивними

потоками подій, і створює основу для подальшої апробації інформаційно-аналітичної системи в реальних умовах експлуатації.

3.4. Апробація створеної інформаційно-аналітичної системи та результати

3.4.1. Створення проєкту та експерименту

Для проведення експериментального A/B-тестування спершу було створено новий проєкт у системі. Це здійснено через інтерфейс платформи (Dashboard → Projects → Create New Project). У формі створення проєкту зазначено назву «E-Commerce Demo Store» та опис (наприклад, *«Оптимізація конверсій на демо-сайті електронної комерції»*). Після збереження проєкту система згенерувала унікальний ідентифікатор проєкту та API-ключ для подальшої інтеграції. На рисунку 3.4 зображено інтерфейс створення нового проєкту з заповненими полями назви та опису та модальне вікно з API ключем на рисунку 3.5.

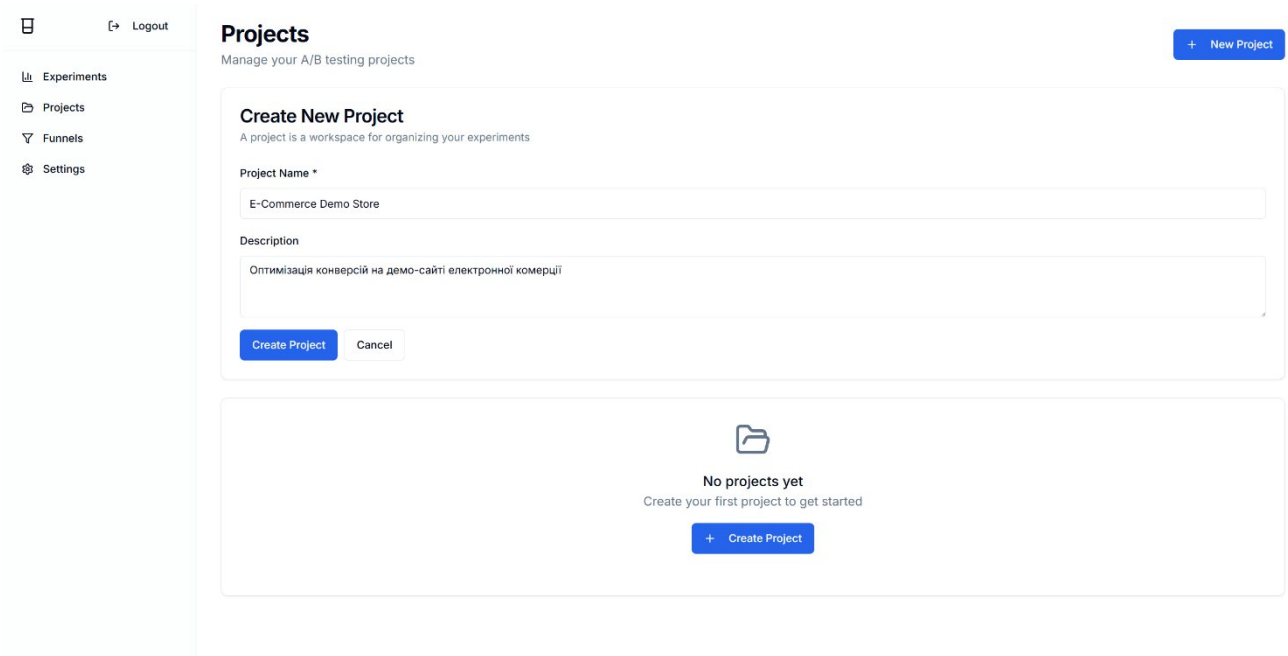


Рисунок 3.4 – Інтерфейс створення нового проєкту

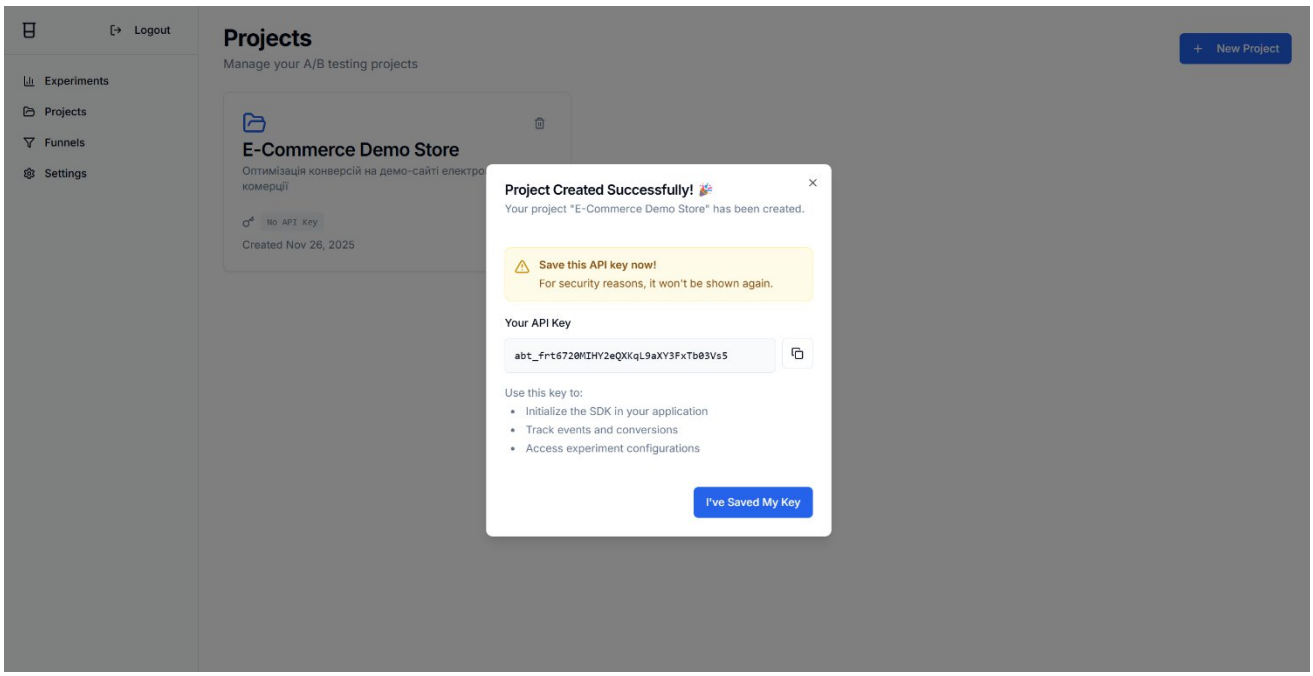


Рисунок 3.5 – Модальне вікно з API ключем створеного проєкту

Наступним кроком у межах створеного проєкту було налаштування експерименту «Оптимізація кнопки “Додати в кошик”». Експеримент створювався через меню платформи (Dashboard → Experiments → Create New Experiment), що відкриває майстер налаштування з послідовними кроками. На етапі базової інформації було задано назву експерименту, його опис та гіпотезу. Зокрема, гіпотеза сформульована так: *«Зміна тексту кнопки на “Buy Now” або зміна її кольору на зелений підвищить конверсію додавання в кошик не менше ніж на 15%.»* Вказану гіпотезу планується перевірити, порівнявши контрольний варіант кнопки із двома тестовими варіантами.

На етапі конфігурації варіантів було створено три варіанти інтерфейсу кнопки:

1. Control,
2. Variant A,
3. Variant B.

Трафік користувачів розподілено між варіантами майже порівну: 34% всіх відвідувачів бачитимуть контрольну версію, а по 33% – кожен з тестових варіантів. У таблиці 3.4 узагальнено налаштування варіантів експерименту.

Таблиця 3.4 – Варіанти експерименту «Оптимізація кнопки “Додати в кошик”»

Варіант	**Зміни в інтерфейсі**	**Частка трафіку**
Control	Початковий вигляд кнопки: текст *“Add to Cart”*, фіолетовий фон (#6366f1)	34%
Variant A	Заміна тексту кнопки на *“Buy Now”*	33%
Variant B	Заміна кольору фону кнопки на зелений (#10b981)	33%

Далі було визначено цілі експерименту для відстеження конверсії. Основною метрикою успіху обрано подію натискання кнопки *“Add to Cart”*, оскільки метою є збільшення саме цих кліків (додавань товару в кошик). Ця подія налаштована як Primary Goal типу *custom event* з назвою події add_to_cart. Додатково налаштовано другорядну ціль (Secondary Goal) – перегляд сторінки кошика після додавання товару, що є подією типу *page_view* з назвою cart_page_view. Таким чином, система фіксуватиме не лише факт кліку по кнопці, а й подальший перехід користувача до кошика, що дозволяє будувати воронку конверсії. У таблиці 3.5 наведено конфігурацію цілей експерименту.

Таблиця 3.5 – Налаштовані цілі експерименту

Ціль	**Тип метрики**	**Подія (event name)**	**Первинна?**
Add to Cart Conversion (натискання «Додати в кошик»)	custom event	add_to_cart	Так (Primary)
Proceed to Cart (перехід до кошика)	page view	cart_page_view	Ні (Secondary)

На наступному кроці було задано таргетинг експерименту – сторінки, на яких проводиться тест. В даному випадку цільовою сторінкою є сторінка товару, тому URL-адресу задано як шаблон, що містить рядок /product.html. Це означає,

що експеримент буде активуватися на всіх сторінках, URL яких містить /product.html (тобто на сторінках окремих товарів демо-сайту). В полі розподілу трафіку (Traffic Allocation) встановлено значення 100%, тобто всі відвідувачі цільової сторінки беруть участь в експерименті (розподіляються між варіантами згідно ваг, вказаних у табл. 3.4).

На фінальному кроці налаштувань експерименту були задані додаткові параметри експерименту для коректності та достовірності результатів. Зокрема, увімкнено опцію Sticky Bucketing, яка гарантує, що кожен унікальний користувач завжди бачитиме один і той самий варіант (його приналежність до варіанту фіксується при першому попаданні в експеримент). Для тестування мінімальний розмір вибірки (Sample Size) встановлено на рівні 10 користувачів – тобто експеримент має зібрати принаймні 10 відвідувачів на варіант перед тим, як можна буде робити статистично обґрунтовані висновки. Поріг статистичної значущості (Significance Threshold) обрано 95% (p = 0,05), що є стандартним рівнем для підтвердження ефекту. Основні параметри таргетингу та налаштувань експерименту наведено у таблиці 3.6.

Таблиця 3.6 – Таргетинг та параметри експерименту

Параметр	**Значення**
Цільова сторінка	URL містить /product.html (сторінка товару)
Розподіл трафіку	100% (всі відвідувачі сторінки)
Sticky bucketing	Увімкнено (фіксація варіанта за користувачем)
Мінімальний розмір вибірки	10 користувачів на варіант
Поріг значущості	95% (p = 0,05)

Після заповнення усіх зазначених налаштувань експеримент було збережено у системі. На рисунках 3.6 та 3.7 показано інтерфейс налаштованого експерименту.

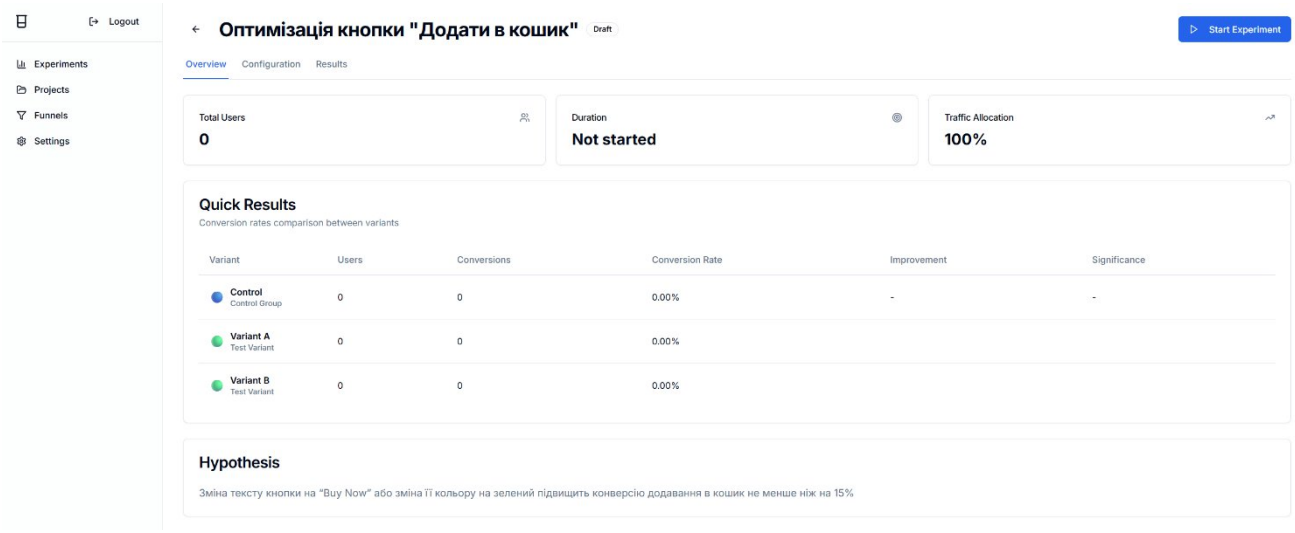


Рисунок 3.6 – Сторінка огляду експерименту

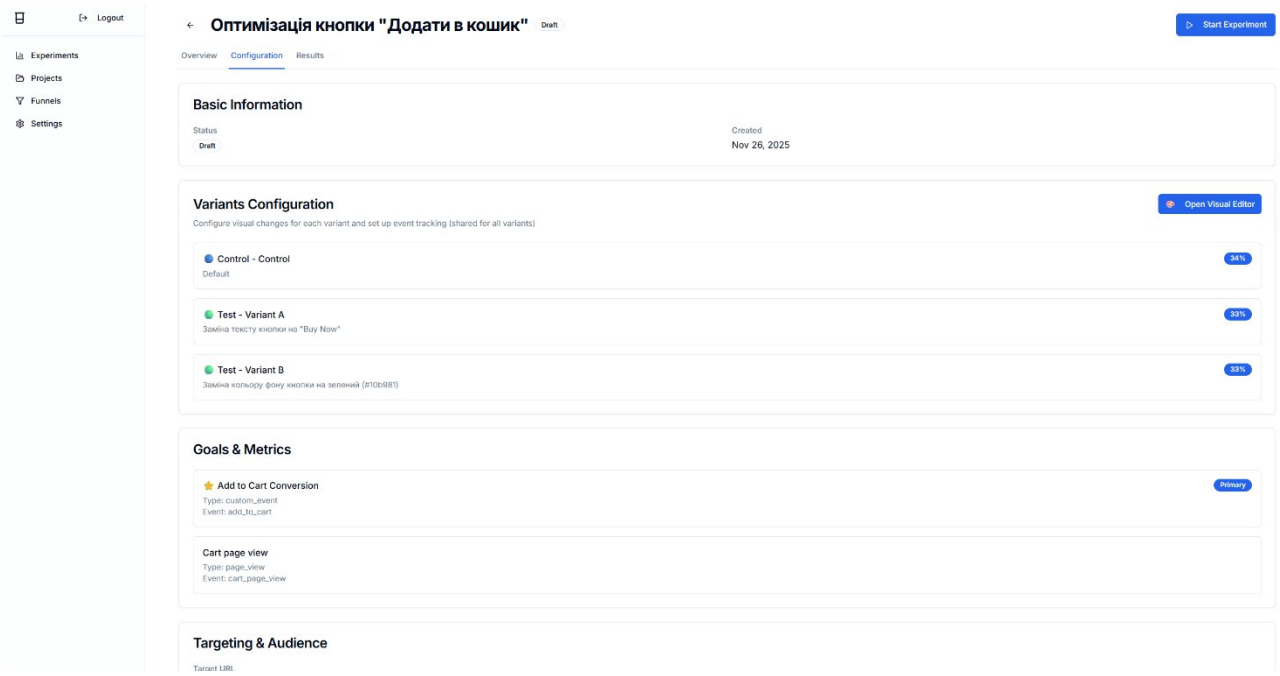


Рисунок 3.7 – Сторінка конфігурації експерименту

На цьому етапі статус створеного експерименту знаходиться в стані *«Draft»* (чернетка) – експеримент готовий до запуску, але ще не активований для користувачів.

3.4.2. Налаштування варіантів експерименту у візуальному редакторі

Для проведення експерименту з оптимізації кнопки «Додати в кошик» на сторінці товару було підготовлено два тестових варіанти. Зокрема, Variant A

передбачає зміну тексту кнопки з *«Add to Cart»* на *«Buy Now»*, а Variant B – зміну кольору фону кнопки з початкового фіолетового на зелений, при тому контрольний варіант залишено без змін. Таблиця 3.7 узагальнює параметри кожного варіанта.

Таблиця 3.7 – Варіанти експерименту для кнопки «Додати в кошик»

Варіант	**Текст на кнопці**	**Колір фону кнопки**	**Опис змін**
Контрольний	Add to Cart	#6366f1 (базовий)	Без змін (оригінальний вигляд)
Variant A	Buy Now	#6366f1 (не змінювався)	Заміна тексту на більш закликальний
Variant B	Add to Cart	#10b981 (зелений)	Зміна кольору фону кнопки на помітніший

Перед внесенням цих змін через інтерфейс платформи користувач відкриває візуальний редактор для створеного експерименту. На рисунку 3.8 показано екран візуального редактора після завантаження цільової сторінки продукту з початковою кнопкою «Add to Cart» (контрольний варіант).

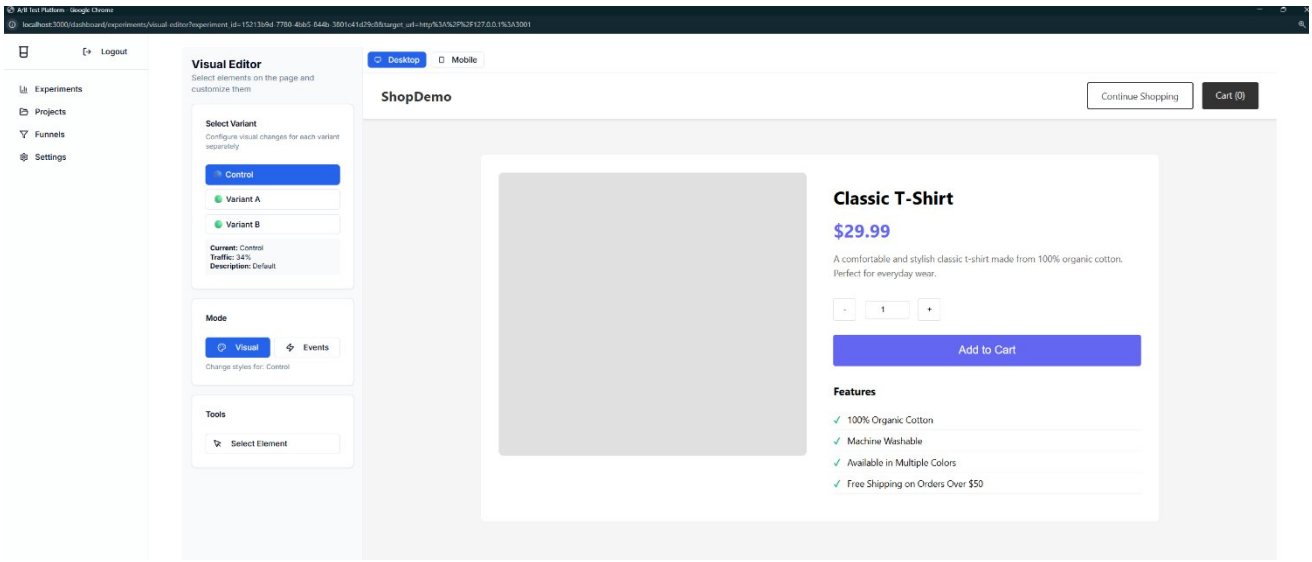


Рисунок 3.8 – Інтерфейс візуального редактора для експерименту (початковий стан сторінки з кнопкою «Додати в кошик»)

Інтерфейс редактора містить панель інструментів із перемикачем варіантів та режимів редагування. У списку варіантів відображено контрольний і тестові варіанти, між якими можна швидко перемикатися. Також доступні два режими роботи: візуальний (для редагування стилів та вмісту) і режим подій (для налаштування відстеження подій). Таким чином, одне вікно редактора дозволяє послідовно налаштувати всі варіанти експерименту без написання коду.

Першим кроком налаштування є вибір варіанта для редагування у випадаючому списку. Для внесення змін тексту обирається Variant A, після чого активується візуальний режим редагування. Далі в області попереднього перегляду користувач клікає на кнопку «Add to Cart», щоб вибрати цей елемент для зміни (при наведенні елемент підсвічується рамкою).

У випадку варіанта А основна зміна – текст кнопки. Для цього у редакторі контенту вводиться новий текст – «Buy Now», який одразу замінює собою оригінальний напис на кнопці у вікні попереднього перегляду. В панелі змін відображається додана зміна властивості textContent для селектора кнопки. На рисунку 3.9 показано результат: текст кнопки змінено на «Buy Now» у варіанті А.

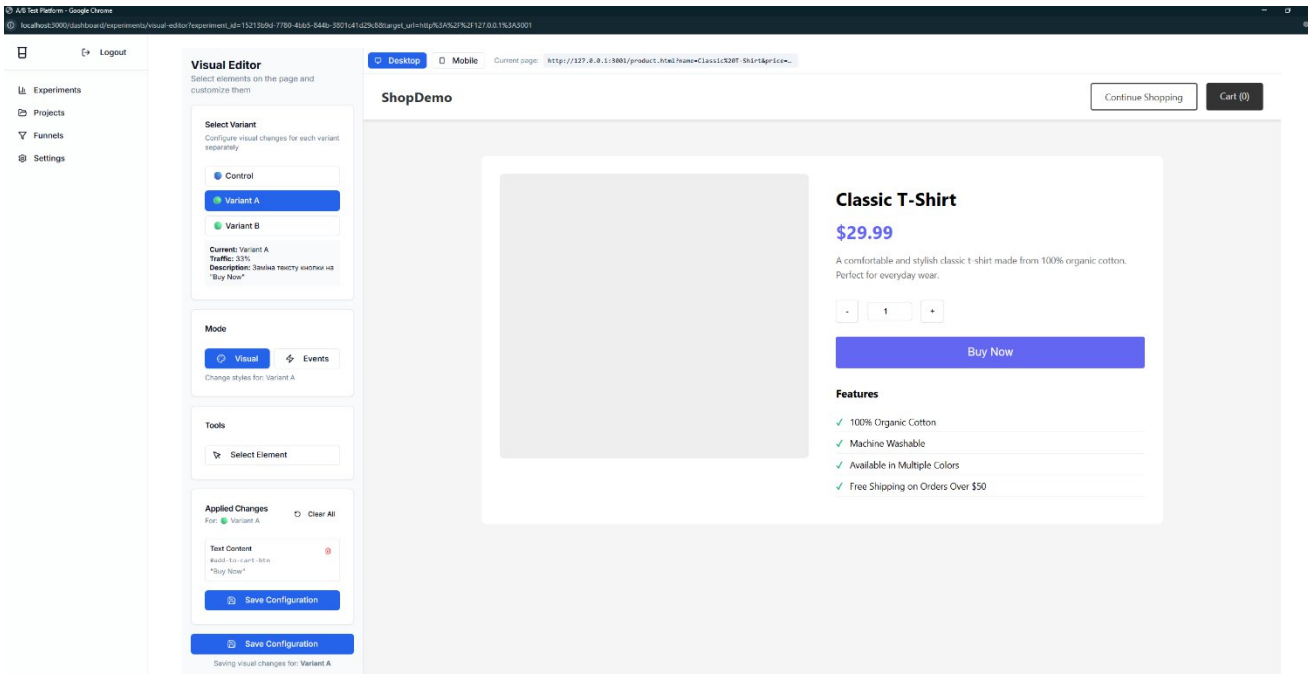


Рисунок 3.9 – Зміна тексту кнопки на «Buy Now» для варіанта A

Як видно з рисунку 3.9, візуальний редактор застосував новий текст до вибраного елемента без перезавантаження сторінки. Це демонструє можливості редактора контенту: текстові вузли можна редагувати прямо на сторінці, що дає змогу оперативно перевіряти різні формулювання CTA. Після цього система автоматично додає цю правку до конфігурації варіанта A (збережено селектор елемента та нове текстове значення).

Наступним етапом є налаштування Variant B, який передбачає зміну стилю кнопки. Експериментатор перемикається на варіант B у тому ж вікні редактора – сторінка перезавантажується до початкового стану (для цього варіанта ще нема змін). Аналогічно до попереднього кроку, користувач обирає кнопку «Add to Cart» на сторінці, але цього разу використовує редактор стилів. На рисунку 3.10 показано процес зміни кольору фону кнопки на яскраво-зелений для варіанта B.

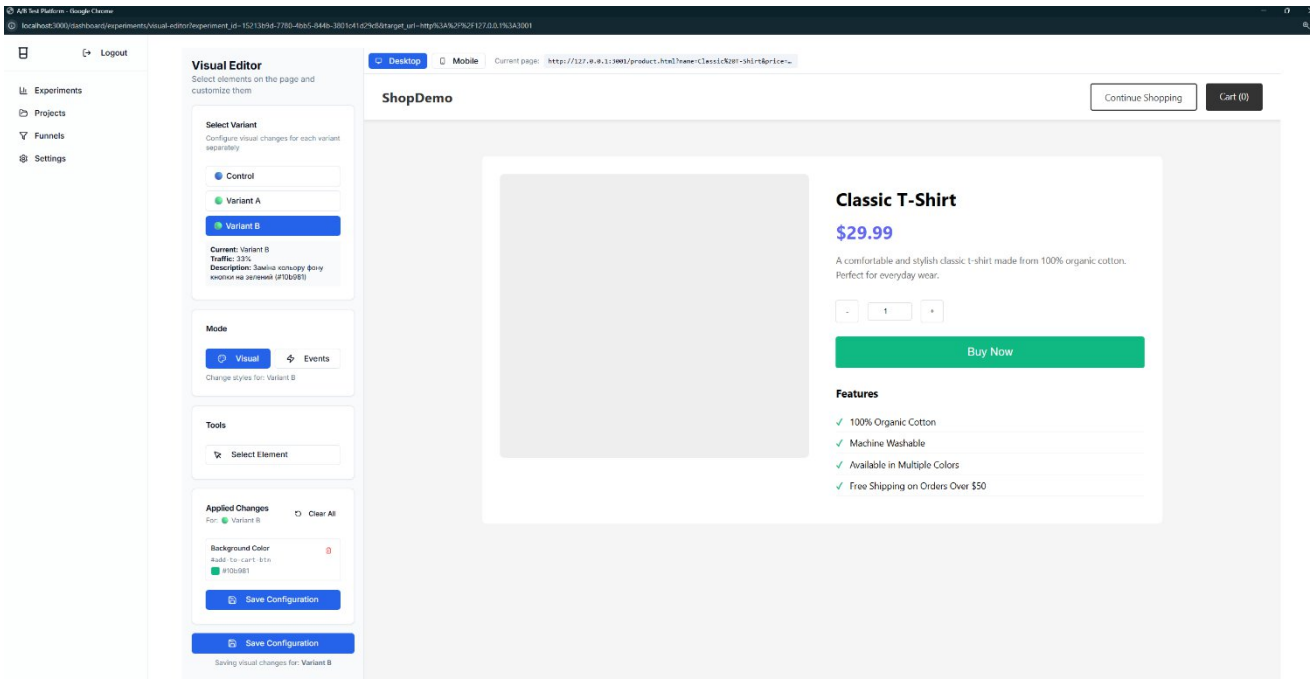


Рисунок 3.10 – Налаштування кольору фону кнопки (зелений) у візуальному редакторі для варіанта B.

Редактор стилів на правій панелі дозволяє змінювати CSS-властивості вибраного елемента. У випадку варіанта B було змінено властивість фону кнопки на зелений відтінок #10b981. В результаті кнопка у вікні попереднього перегляду одразу відображається зеленого кольору. Ця стилістична зміна також заноситься до списку змін варіанта B – зберігається CSS-селектор #add-to-cart-btn та нове значення властивості background (зелений колір).

Окрім візуальних змін, в редакторі можна налаштувати й відстеження подій (режим «Events»). Для даного експерименту цільовою метрикою виступає зміна значення поля для введення кількості товару, тому в режимі подій користувач додає відстежування «Value Change» на цей елемент та присвоює назву (напр., pdp_quantity_change). На рисунку 3.11 показано конфігурацію відстеження кліку для кнопки в режимі подій.

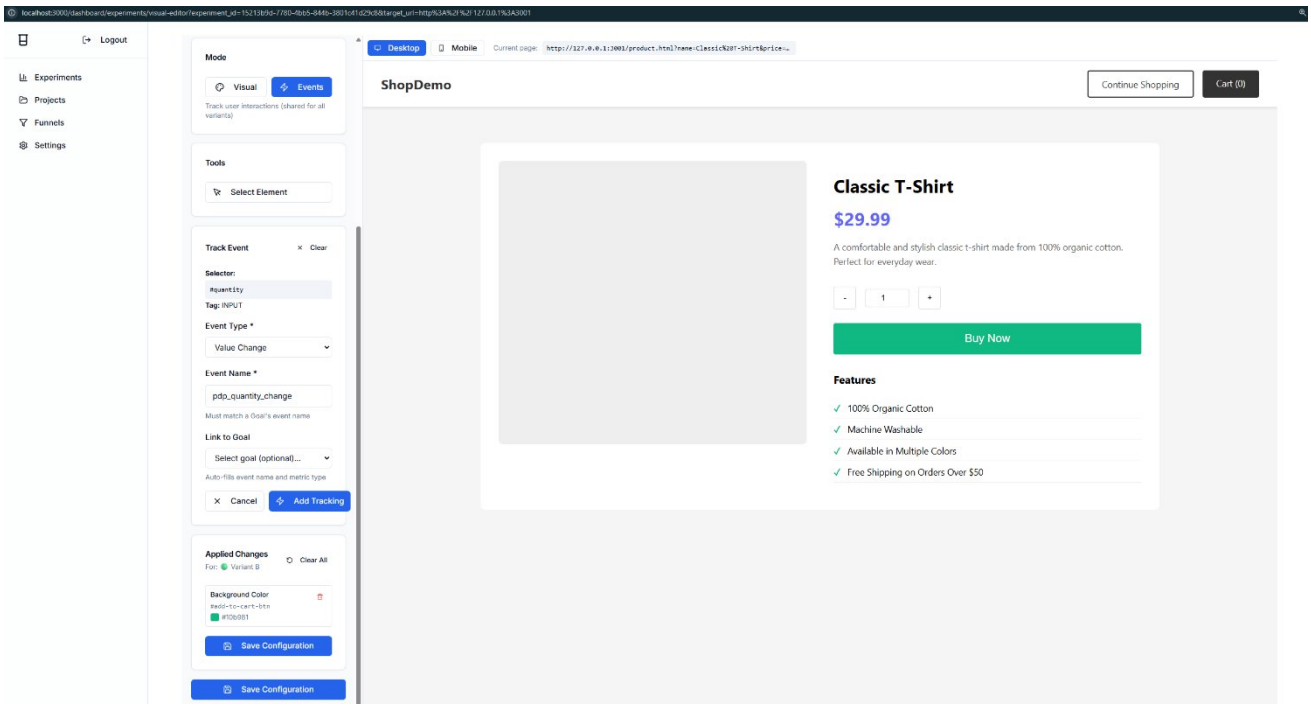


Рисунок 3.11 – Додавання відстеження зміни кількості

Усі події, додані в цьому режимі, позначаються як “спільні для всіх варіантів” – тобто незалежно від варіанту, натискання кнопки буде фіксуватися системою. Налаштована подія одразу додається до конфігурації експерименту і в подальшому автоматично відслідковуватиметься на сайті SDK-бібліотекою.

Завершивши внесення змін для обох варіантів, користувач натискає кнопку «Зберегти» у візуальному редакторі. Після цього усі задані модифікації серіалізуються у форматі JSON і зберігаються в конфігурації експерименту. Зокрема, для кожного варіанта формується список об’єктів змін із вказанням CSS-селектора цільового елемента, назви змінюваного атрибута та нового значення. Наприклад, для Variant A генерується запис про зміну textContent кнопки на *«Buy Now»*, а для Variant B – зміна background на *#10b981*. Ця інформація зберігається у полі variant.config.changes відповідного варіанта та надалі використовується на стороні клієнта. Коли користувач потрапляє на сторінку товару, SDK надсилає запит на призначення варіанту, отримує конфігурацію експерименту і автоматично застосовує візуальні зміни відповідно до отриманих селекторів та стилів, а також відслідковує додані події.

3.4.3. Налаштування інтеграції та відстеження подій

3.1

3.2

3.3

3.4

3.4.1

3.4.2

3.4.3

3.4.3.1. Інтеграція SDK

Для проведення тесту на реальній сторінці необхідно інтегрувати експеримент у фронтенд демо-сайту. З цією метою сайт було додано підключення клієнтського SDK платформи A/B-тестування. Фрагмент коду інтеграції наведено у лістингу 3.1. В цьому коді виконуються такі дії: ініціалізація SDK з використанням виданого API-ключа та URL-ендпоїнта серверу, отримання варіанту експерименту для поточного користувача, автоматичне застосування змін інтерфейсу згідно з конфігурацією варіанта, а також встановлення обробника події кліку по кнопці «Add to Cart» для відправки інформації про додавання в кошик в систему аналітики.

Лістинг 3.1 – Фрагмент коду інтеграції експерименту на сторінці товару

```
<script src="http://localhost:3000/ab-test-sdk.js"></script>
<script>
  // Ініціалізація SDK з API-ключем та ендпоїнтом
  const sdk = new ABTestSDK({
    apiKey: 'abt_eEAisLnugcTWzVNlfUEnvE6fNPfWIVWF',  // API-ключ проєкту
    endpoint: 'http://localhost:3000/api',
    debug: true  // режим відладки для логування
```

```
  });
  // Після успішної ініціалізації SDK...
  sdk.init().then(() => {
    const experimentId = 'b91cdedb-c482-4364-9106-01c190b70bfa'
    const variant = sdk.getVariant(experimentId);
    // SDK застосовує зміни варіанта до елементів сторінки та автоматично відстежує подію перегляду сторінки (page_view)
    // Відстеження події "Add to Cart"
    document.getElementById('add-to-cart-btn').addEventListener('click', () => {
      // Бізнес-логіка додавання товару в кошик
      const cart = JSON.parse(localStorage.getItem('cart') || '[]');
      cart.push({ name: productName, price: productPrice });
      localStorage.setItem('cart', JSON.stringify(cart));
      // Відправка події "add_to_cart" до системи аналітики
      sdk.track('add_to_cart', {
        product: productName,
        product_id: productId,
        price: productPrice,
        cart_count: cart.length,
        experiment_id: experimentId,
        variant_id: variant?.variant_id
      });
    });
  });
</script>
```

Як видно з наведеного коду, після ініціалізації sdk.init() здійснюється виклик sdk.getVariant(experimentId). Ця функція визначає, який саме варіант експерименту має бути показано поточному відвідувачу сторінки. SDK використовує детермінований алгоритм розподілу користувачів (на основі

хешування і user_id), тому кожен користувач закріплюється за певним варіантом згідно встановлених ваг (sticky bucketing). Якщо користувач вже брав участь в експерименті раніше, він отримає той самий варіант, що і при першому відвідуванні, завдяки збереженню інформації в localStorage. Після визначення варіанта SDK автоматично застосовує зміни інтерфейсу, описані в конфігурації цього варіанта (наприклад, змінює текст кнопки або її колір). Таким чином, відвідувачі почергово бачитимуть різні версії кнопки додавання в кошик відповідно до налаштувань експерименту, якщо він запущений.

3.4.3.2. Запуск експерименту

Після інтеграції коду на сторінці експеримент було запущено через інтерфейс адміністратора (Dashboard → Experiments → [Обраний експеримент] → Start Experiment). Зміна статусу експерименту на «running» означає, що з цього моменту всі користувачі, які відкриють сторінку товару, автоматично потрапляють в експеримент і побачать один з варіантів кнопки. Перед запуском експеримент мав статус «Draft», тож його результати не збиралися і зміни на сайті не застосовувалися. Переконавшись, що інтеграцію виконано правильно і SDK підключено, експеримент переведено в активний режим.

На рисунку 3.12 зображено приклад відображення кнопки на сторінці товару для тестового варіанта: у варіанті A текст кнопки змінено на «Buy Now».

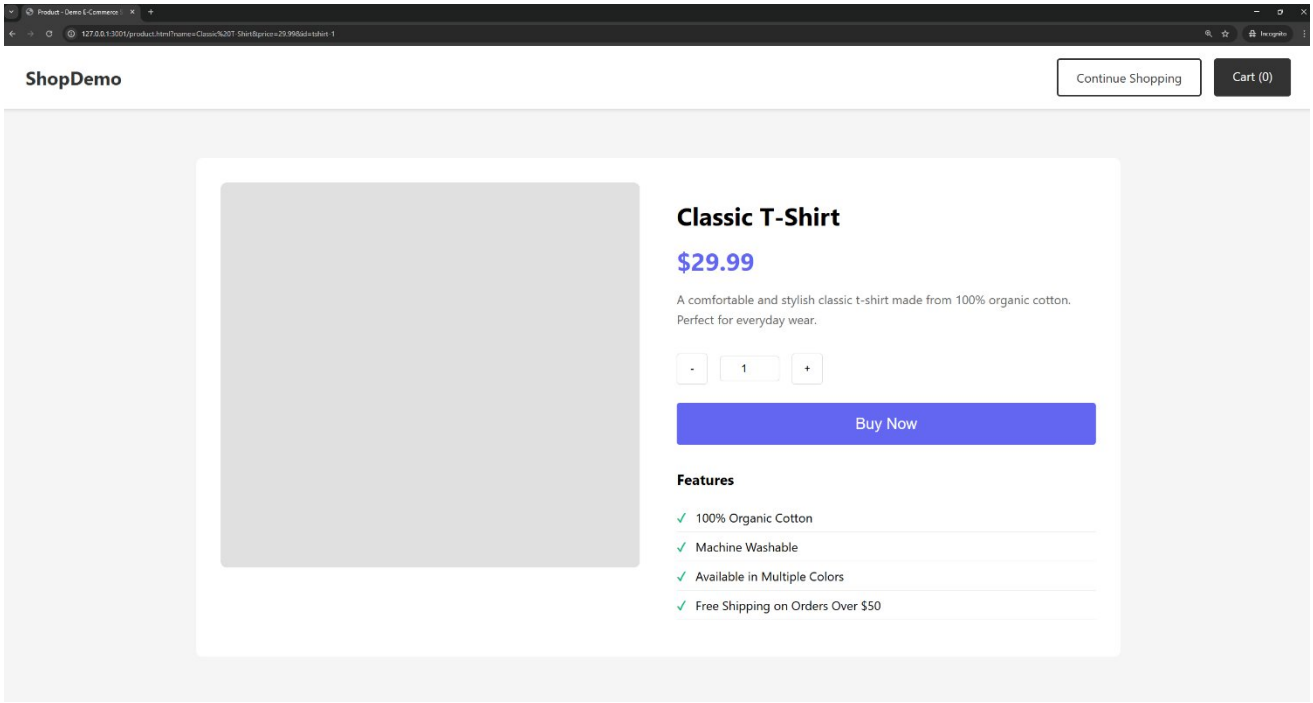


Рисунок 3.12 – Змінений інтерфейс кнопки «Add to Cart» для Variant A

3.4.3.3. Відстеження подій та збір даних

Після запуску експерименту система почала збір аналітичних даних про поведінку користувачів. SDK у браузері автоматично реєструє подію перегляду сторінки при завантаженні продуктового шаблону (імпресія варіанту, подія типу page_view). Це означає, що кожен візит на сторінку товару фіксується як показ експерименту користувачу, з прив’язкою до конкретного варіанта. Крім того, при взаємодії користувача з кнопкою «Add to Cart» викликається метод *sdk.track('add_to_cart', {...})*, що відправляє на сервер подію типу **custom event** з назвою *add_to_cart*. Ця подія містить додаткові параметри (такі як ідентифікатор товару, назва товару, ціна, кількість товарів у кошику тощо, згідно з листингом 3.1) і слугує сигналом конверсії для експерименту. Таким чином, для кожного варіанта система збирає дві основні метрики: кількість показів сторінки (impressions) та кількість кліків «Add to Cart» (conversions). Паралельно фіксується

і другорядна метрика – переходи на сторінку кошика (cart_page_view), якщо такі відбуваються після додавання товару.

В процесі збору даних важливо гарантувати, що кожен користувач рахується тільки один раз для кожного варіанта і що повторні дії враховуються коректно. Завдяки sticky bucketing один користувач не «стрибає» між варіантами, а отже, його дії (перегляд сторінки та можливий клік) пов’язуються з одним конкретним варіантом. SDK зберігає призначення варіанта в локальному сховищі браузера та на бекенді, що забезпечує цілісність даних експерименту.

3.4.4. Побудова воронки та аналіз поведінки користувачів

Для комплексного аналізу впливу змін інтерфейсу на поведінку користувачів було сформовано воронку взаємодії, яка відображає послідовність ключових подій на шляху від перегляду товару до оформлення покупки. Створення такої воронки дозволяє оцінити не лише безпосередній ефект від зміни кнопки «Додати в кошик», але й подальший вплив експериментальних варіантів на глибші етапи конверсійного шляху.

3.4.3.4. Створення нової воронки

У панелі адміністратора системи було створено нову воронку, яка включає п’ять основних етапів користувацької взаємодії. На рисунку 3.13 наведено інтерфейс створення воронки з визначенням кожного з кроків.

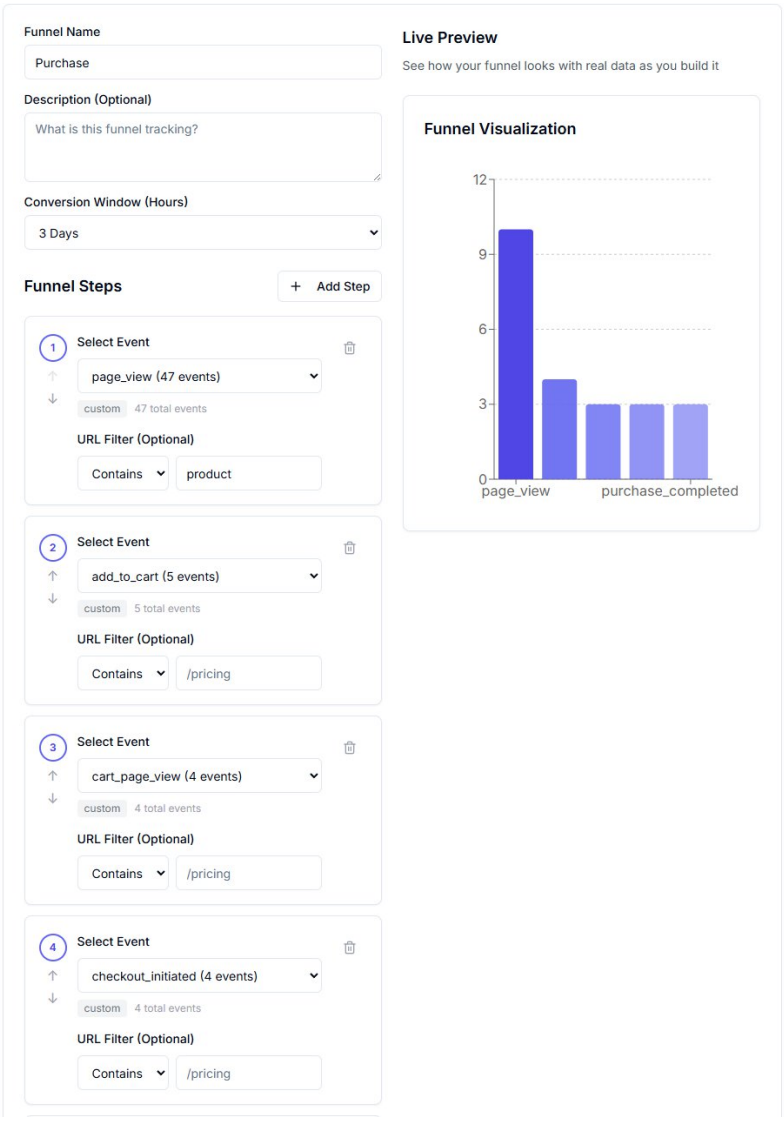


Рисунок 3.13 – Інтерфейс створення воронки у системі аналітики

На рисунку показано форму додання нової воронки з переліком послідовних кроків. Кожен із кроків воронки відображає окрему подію, яка фіксується системою через SDK:

Таблиця 3.8 – Кроки воронки

Крок	**Назва події**	**Опис**
1	page_view	Перегляд сторінки товару
2	add_to_cart	Додавання товару в кошик (клік по кнопці)

Закінчення таблиці 3.8

Крок	Назва події	Опис
3	cart_page_view	Перегляд кошика
4	checkout_initiated	Початок оформлення замовлення
5	purchase_completed	Успішне завершення покупки

У воронці було також задано часовий інтервал 3 дні для проходження користувачем усіх кроків. Це дозволяє оцінювати як швидкі конверсії, так і відкладені дії, типові для поведінки реальних відвідувачів інтернет-магазину.

3.4.3.5. Параметри фільтрації та прив’язка воронки до експерименту

Після створення воронки було застосовано фільтри за експериментом та варіантами. У вкладці Filters обрано:

- Experiment: «Оптимізація кнпоки “Додати в кошик”»
- Variant: Control / Variant A / Variant B

Це дозволяє аналізувати проходження кожного етапу воронки окремо для кожної групи користувачів. На рисунках 3.14 та 3.15 наведено приклад застосованих фільтрів для прив’язки воронки до експерименту.

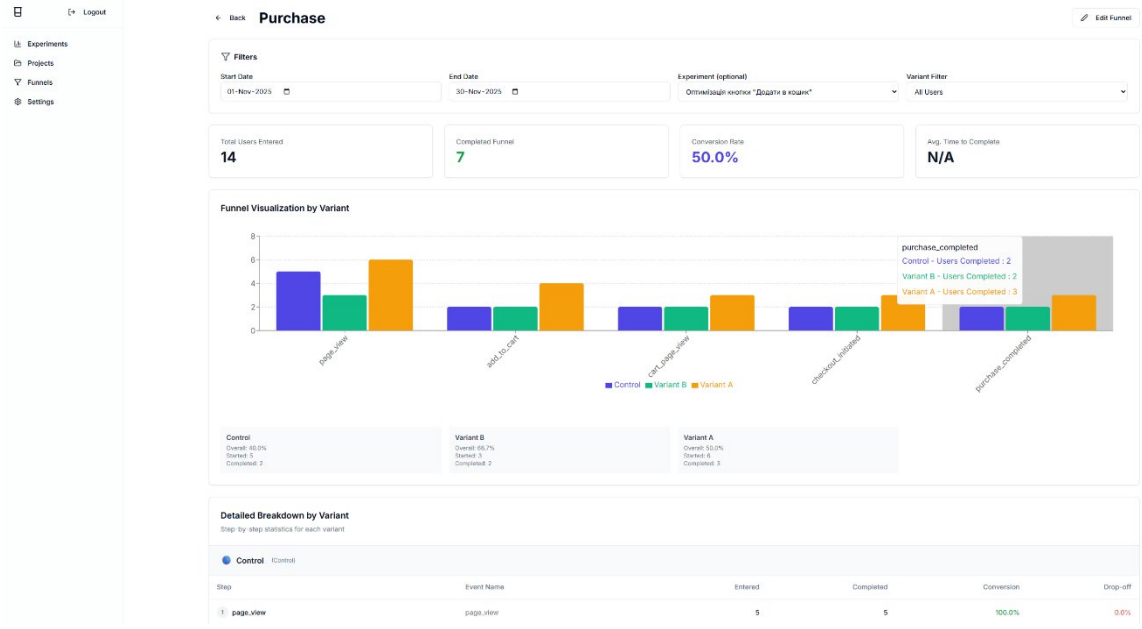


Рисунок 3.14 – Фільтрація воронки за експериментом та всіма варіантами

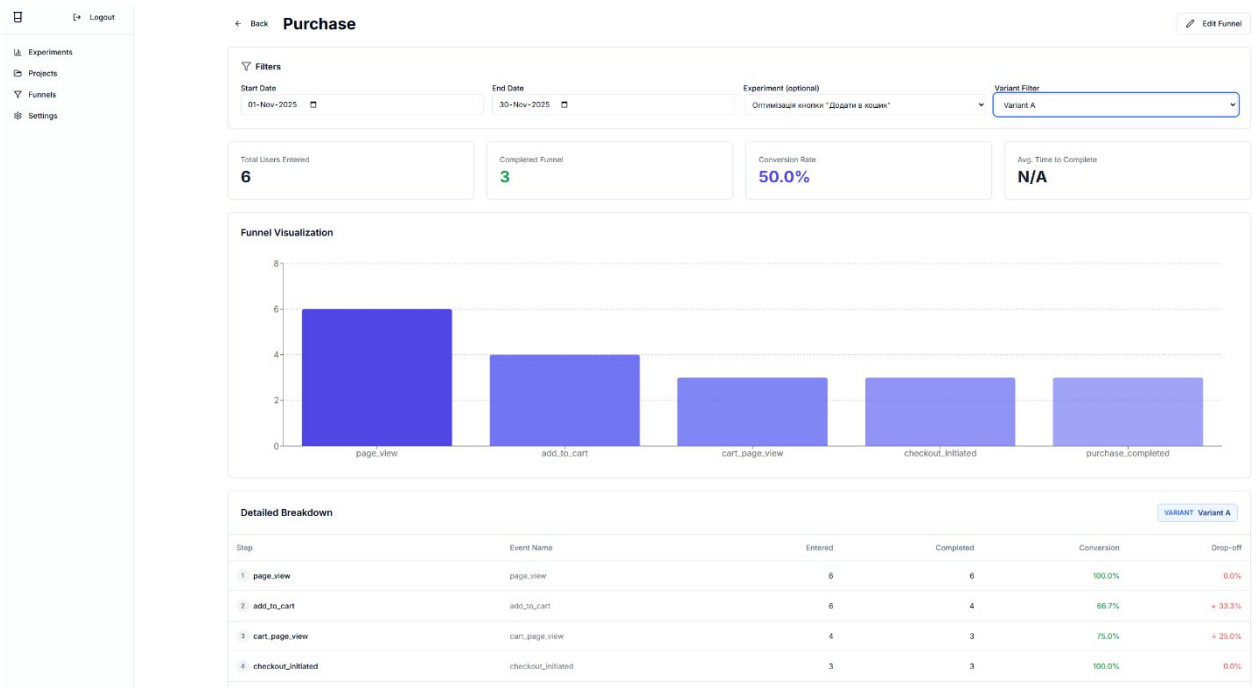


Рисунок 3.15 – Фільтрація воронки за експериментом та варіантом A

Після застосування фільтрів аналітичний модуль системи будує воронку, де відображається кількість користувачів на кожному етапі та відсоток переходу до наступного кроку.

У результаті система формує такі показники для кожного варіанта:

1. Conversion Rate між етапами ($CR_1 \rightarrow CR_5$)

   Наприклад:

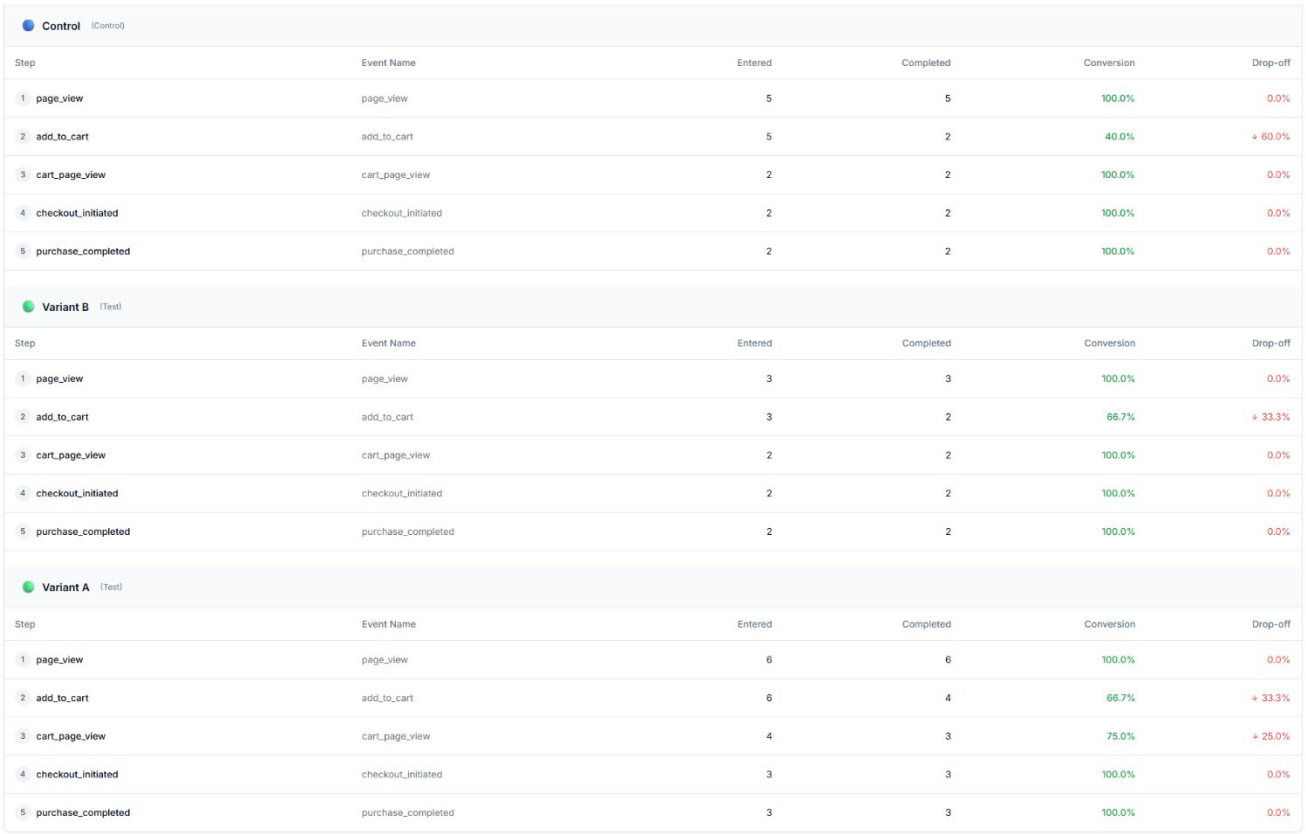
   - Product View → Add to Cart
   - Add to Cart → Cart View
   - Cart View → Checkout
   - Checkout → Purchase Completed

2. Drop-off (%) — частка користувачів, які припинили взаємодію на конкретному етапі.

3. Загальна конверсія воронки — частка відвідувачів, які успішно завершили покупку.

3.4.3.6. Порівняння варіантів за показниками воронки

Для кожного варіанта експерименту (Control, Variant A, Variant B) система побудувала окрему воронку переходів. На рисунку 3.16 наведено фрагмент отриманої таблиці, що демонструє значення для кожного кроку: кількість користувачів, які увійшли на етап (Entered), завершили його (Completed), а також розраховану конверсію та drop-off.



Control (Control)

Step	Event Name	Entered	Completed	Conversion	Drop-off
1 page_view	page_view	5	5	100.0%	0.0%
2 add_to_cart	add_to_cart	5	2	40.0%	↓ 60.0%
3 cart_page_view	cart_page_view	2	2	100.0%	0.0%
4 checkout_initiated	checkout_initiated	2	2	100.0%	0.0%
5 purchase_completed	purchase_completed	2	2	100.0%	0.0%

Variant B (Test)

Step	Event Name	Entered	Completed	Conversion	Drop-off
1 page_view	page_view	3	3	100.0%	0.0%
2 add_to_cart	add_to_cart	3	2	66.7%	↓ 33.3%
3 cart_page_view	cart_page_view	2	2	100.0%	0.0%
4 checkout_initiated	checkout_initiated	2	2	100.0%	0.0%
5 purchase_completed	purchase_completed	2	2	100.0%	0.0%

Variant A (Test)

Step	Event Name	Entered	Completed	Conversion	Drop-off
1 page_view	page_view	6	6	100.0%	0.0%
2 add_to_cart	add_to_cart	6	4	66.7%	↓ 33.3%
3 cart_page_view	cart_page_view	4	3	75.0%	↓ 25.0%
4 checkout_initiated	checkout_initiated	3	3	100.0%	0.0%
5 purchase_completed	purchase_completed	3	3	100.0%	0.0%

Рисунок 3.16 – Показники проходження воронки для варіантів Control, Variant A та Variant B

Аналіз даних показує такі результати:

Control:

- 5 користувачів переглянули сторінку товару, всі 5 перейшли до кліку «Add to cart».
- Конверсія з перегляду товару у додавання в кошик — 40% (2 з 5).
- Усі користувачі, які додали товар, довели покупку до кінця (100%).

Variant B (зміна кольору кнопки):

- 3 перегляди товару → 3 клацання по кнопці «Add to cart».
- Конверсія першого переходу — 66,7%, що на 26,7 п.п. вище за Control.
- Далі Variant B демонструє повну конверсію на всіх етапах (100%).

Variant A (зміна тексту кнопки):

- 6 переглядів товару → 6 кліків по «Add to cart».
- Конверсія першого переходу — 66,7% (аналогічно Variant B).
- Далі:
    - Cart page view → 3 завершення з 4 (75%);
    - Checkout → 100%;
    - Purchase → 100%.

Детальну інформацію про результати проходження воронки подано в таблиці 3.9.

Таблиця 3.9 – Результати проходження воронки та A/B тесту

Етап	**Control**	**Variant B**	**Variant A**
Product view → Add to cart	40%	**66,7%**	**66,7%**
Add to cart → Cart view	100%	100%	75%
Cart view → Checkout	100%	100%	100%
Checkout → Purchase	100%	100%	100%

Можна зробити висновок, що обидва тестові варіанти значно підвищують кількість кліків по кнопці, тобто впливають саме на той етап, який мав бути покращений у рамках експерименту.

Таким чином було здійснено повний цикл розроблення та апробації створеної інформаційно-аналітичної системи A/B-тестування, починаючи від проєктування архітектури та побудови сховища даних і завершуючи практичною перевіркою ефективності системи на реальному прикладі оптимізації елементів інтерфейсу веб-ресурсу.

На етапі проєктування системи було обґрунтовано вибір архітектурного підходу, що включає серверний модуль для управління експериментами, сховища

даних (PostgreSQL та аналітичні таблиці), а також клієнтський SDK для фіксації подій і автоматичного застосування варіантів інтерфейсу. Побудовано логічну модель даних, яка охоплює сутності експериментів, варіантів, подій і конверсій, що забезпечує цілісність і структурованість інформації, необхідної для проведення A/B-тестування.

У підрозділі, присвяченому завантаженню та обробці даних, реалізовано механізм ETL/ELT-процесів: події, отримані від клієнтських застосунків через SDK, проходили валідацію, нормалізацію та зберігалися у сховищі у форматі, придатному для подальшого аналізу. Це забезпечило коректність подальших статистичних обчислень і можливість узгодженого побудування воронок.

Під час розроблення клієнтського SDK та API-шару було реалізовано:

- механізм призначення варіантів користувачам (bucketing);
- збір подій взаємодії в реальному часі;
- обробку конфігурацій експериментів;
- застосування змін інтерфейсу на стороні клієнта без залучення розробника.

Особливу увагу приділено візуальному редактору варіантів, що забезпечує можливість внесення змін в інтерфейс без написання коду. У межах апробації доведено, що редактор дозволяє швидко формувати альтернативні UI-версії (зміна тексту та стилю кнопки) та автоматично серіалізує внесені зміни у конфігурацію експерименту.

На завершальному етапі здійснено апробацію системи шляхом проведення експерименту з оптимізації кнопки «Додати в кошик» на сторінці товару. Порівняльний аналіз поведінкових даних по варіантах показав, що внесені зміни позитивно вплинули на ключовий етап переходу «перегляд товару → додавання в кошик». Побудована користувацька воронка дозволила простежити вплив змін на подальші етапи, а також підтвердила, що збільшення активності на початковому кроці призводить до зростання кількості завершених покупок.

Загалом, результати апробації підтвердили функціональність, коректність та практичну цінність розробленої системи. Вона дозволяє швидко створювати та запускати A/B-експерименти, аналізувати поведінку користувачів, оцінювати

ефективність змін і формувати обґрунтовані рекомендації для покращення цифрових продуктів. Таким чином, система повністю відповідає поставленій меті дослідження та може бути використана підприємствами для підвищення ефективності веб-ресурсів та оптимізації користувацького досвіду.

3.5. Висновки до розділу 3

У третьому розділі було здійснено комплексну реалізацію інформаційно-аналітичної системи A/B тестування, що охоплює проєктування архітектури, створення сховища подій, розробку ключових програмних компонентів та їх інтеграцію з веб-ресурсами клієнтів. На основі сформованих у попередніх розділах вимог розроблено узгоджену структуру системи, яка об’єднує серверні модулі, клієнтський SDK, адміністративну панель та засоби аналітичної обробки даних. Особливу увагу приділено організації подієво-орієнтованого збору даних та побудові логічної моделі сховища, що забезпечує коректне зберігання інформації про користувацькі дії та варіанти експериментів.

Проведене експериментальне впровадження системи підтвердило її працездатність у реальних умовах роботи з клієнтським веб-ресурсом. Налаштування та запуск тестового проєкту дали змогу перевірити роботу механізмів детермінованого розподілу користувачів, обліку подій, формування воронки та обчислення основних показників ефективності. Застосування системи для аналізу варіантів інтерфейсного елемента показало, що розроблений інструментарій дозволяє оперативно виявляти зміни в поведінці користувачів, визначати проблемні етапи користувацького шляху та обґрунтовувати рішення щодо оптимізації веб-ресурсів.

Таким чином, результати апробації свідчать, що створена інформаційно-аналітична система відповідає поставленим функціональним вимогам, забезпечує повний цикл роботи з A/B експериментами й може бути використана ФОП «Горенко Тарас Анатолійович» для систематичного підвищення ефективності клієнтських веб-ресурсів. Отримані напрацювання формують практичну основу

для подальшого розширення можливостей системи та її інтеграції у внутрішні процеси підприємства.

ВИСНОВКИ

У кваліфікаційній роботі здійснено комплексне дослідження теоретичних і практичних аспектів застосування A/B-тестування як інструменту підвищення ефективності веб-ресурсів, а також розроблено інформаційно-аналітичну систему, що забезпечує повний цикл проведення експериментів — від формування гіпотези до аналізу результатів. У роботі послідовно вирішено поставлену науково-прикладну задачу, яка полягала у створенні програмного комплексу для дослідження впливу інтерфейсних змін на поведінку користувачів і результативність електронної взаємодії.

Проведений аналіз літературних джерел і існуючих рішень дав змогу уточнити принципи та методологічні засади сучасного A/B-тестування, визначити його місце в системі управління веб-аналітикою та сформувати вимоги до інформаційно-аналітичної системи. На основі цих вимог було розроблено архітектурну концепцію програмного рішення, яка поєднує серверний модуль для управління експериментами, клієнтські засоби збору подій, сховище даних для аналітичної обробки та адміністративну панель для взаємодії з системою. Реалізовано логічну модель даних, що забезпечує узгоджене зберігання інформації про користувацьку активність, варіанти інтерфейсу та результати експериментів.

У межах роботи створено інструмент, який підтримує автоматизований розподіл користувачів між експериментальними варіантами, коректне застосування конфігурацій інтерфейсу на стороні клієнта та збір поведінкових даних у режимі реального часу. Запропоновано та реалізовано модуль візуального редагування інтерфейсу, що дозволяє формувати варіанти експериментів без необхідності втручання у програмний код, що є важливою перевагою для фахівців, які не мають технічної підготовки. Побудоване сховище даних забезпечує можливість накопичення та подальшої обробки подій, що формує основу для розрахунку показників взаємодії та ефективності.

Здійснена апробація системи на прикладі оптимізації кнопки «Додати в кошик» на сторінці товару дала змогу перевірити працездатність запропонованого рішення та оцінити його практичну цінність. Побудована воронка користувацьких дій засвідчила, що зміна текстового наповнення або стилістичних характеристик кнопки призвела до зростання первинної конверсії та позитивно позначилася на загальному проходженні ключових етапів взаємодії. Отримані результати підтвердили, що застосування A/B-тестування дає змогу визначати найбільш ефективні інтерфейсні рішення, ґрунтуючись на реальних даних, а впровадження навіть незначних UI-змін може істотно вплинути на кінцеві показники діяльності веб-ресурсу.

Наукова новизна роботи полягає у розробленні інтегрованої архітектури системи A/B-тестування, яка поєднує візуальне редагування інтерфейсу, сценарне відстеження користувацьких подій та автоматизоване застосування експериментальних змін. На відміну від наявних рішень, запропонована система забезпечує можливість повноцінного керування експериментами без залучення програмістів, що розширює сферу потенційного застосування методу A/B-тестування у підприємницькому середовищі. Практичне значення отриманих результатів полягає у можливості використання розробленої системи підприємствами для підвищення ефективності онлайн-комунікацій, оптимізації користувацького досвіду та підвищення результативності маркетингової діяльності.

Результати апробації доводять, що створений інформаційно-аналітичний інструмент може бути основою для подальшого розвитку та розширення функціоналу, зокрема шляхом реалізації багатофакторних експериментів, інтеграції з CRM-системами, удосконалення статистичного модуля та розширення можливостей візуального редактора. Таким чином, поставлена мета роботи досягнута, а отримані результати становлять як теоретичну, так і практичну цінність для сфери веб-аналітики та цифрової економіки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Amplitude Experiment Documentation. Amplitude. URL: https://www.docs.developers.amplitude.com/experiment/ (дата звернення: 25.11.2025).
2. Mixpanel Analytics Documentation. Mixpanel. URL: https://developer.mixpanel.com/ (дата звернення: 25.11.2025).
3. Optimizely A/B Testing Platform. Optimizely. URL: https://www.optimizely.com/ (дата звернення: 25.11.2025).
4. VWO A/B Testing Platform. VWO. URL: https://vwo.com/ (дата звернення: 25.11.2025).
5. LaunchDarkly Feature Flag Platform. LaunchDarkly. URL: https://launchdarkly.com/ (дата звернення: 25.11.2025).
6. Split.io Feature Experimentation Platform. Split. URL: https://www.split.io/ (дата звернення: 25.11.2025).
7. Statsig Experimentation Platform. Statsig. URL: https://statsig.com/ (дата звернення: 25.11.2025).
8. GrowthBook Open-Source A/B Testing. GrowthBook. URL: https://www.growthbook.io/ (дата звернення: 25.11.2025).
9. Google Optimize (Archived Documentation). Google. URL: https://support.google.com/optimize/ (дата звернення: 25.11.2025).
10. Supabase Documentation. Supabase. URL: https://supabase.com/docs (дата звернення: 25.11.2025).
11. PostgreSQL Documentation. PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs/ (дата звернення: 25.11.2025).
12. Prisma ORM Documentation. Prisma Data. URL: https://www.prisma.io/docs (дата звернення: 25.11.2025).
13. ClickHouse Documentation. ClickHouse. URL: https://clickhouse.com/docs (дата звернення: 25.11.2025).

14. Node.js Documentation. Node.js Foundation. URL: https://nodejs.org/en/docs (дата звернення: 25.11.2025).
15. Express.js Web Framework. Express. URL: https://expressjs.com/ (дата звернення: 25.11.2025).
16. TypeScript Language Documentation. Microsoft. URL: https://www.typescriptlang.org/docs/ (дата звернення: 25.11.2025).
17. React Official Documentation. Meta. URL: https://react.dev/ (дата звернення: 25.11.2025).
18. Next.js Documentation. Vercel. URL: https://nextjs.org/docs (дата звернення: 25.11.2025).
19. Redux Toolkit Documentation. Redux. URL: https://redux-toolkit.js.org/ (дата звернення: 25.11.2025).
20. Web Vitals Documentation. Google. URL: https://web.dev/vitals/ (дата звернення: 25.11.2025).
21. REST API Principles. RESTful API Tutorial. URL: https://restfulapi.net/ (дата звернення: 25.11.2025).
22. MDN Web Docs: DOM API. Mozilla Foundation. URL: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model (дата звернення: 25.11.2025).
23. Docker Documentation. Docker Inc. URL: https://docs.docker.com/ (дата звернення: 25.11.2025).
24. GitHub Documentation. GitHub. URL: https://docs.github.com/ (дата звернення: 25.11.2025).
25. Webhooks Guide. Stripe Developers. URL: https://stripe.com/docs/webhooks (дата звернення: 25.11.2025).
26. UI/UX Guidelines. Nielsen Norman Group. URL: https://www.nngroup.com/articles/ (дата звернення: 25.11.2025).
27. Product Analytics Basics. Heap Analytics. URL: https://heap.io/resources (дата звернення: 25.11.2025).

28. Event Tracking Concepts. Segment Documentation. URL: https://segment.com/docs/ (дата звернення: 25.11.2025).

ДОДАТКИ

Додаток А. BPMN-діаграми бізнес-процесів системи

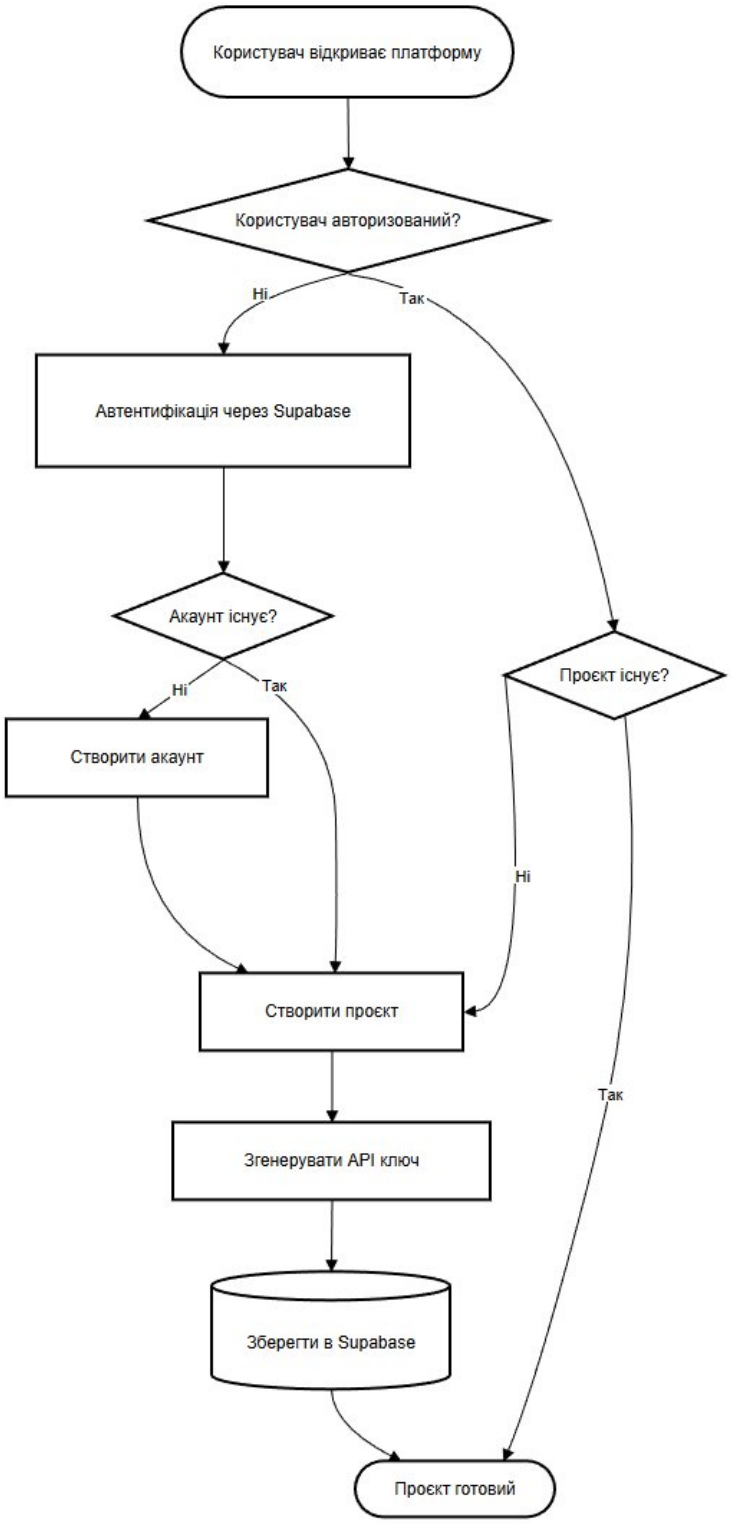


Рисунок А.1 — Процес онбордингу користувача та налаштування проєкту

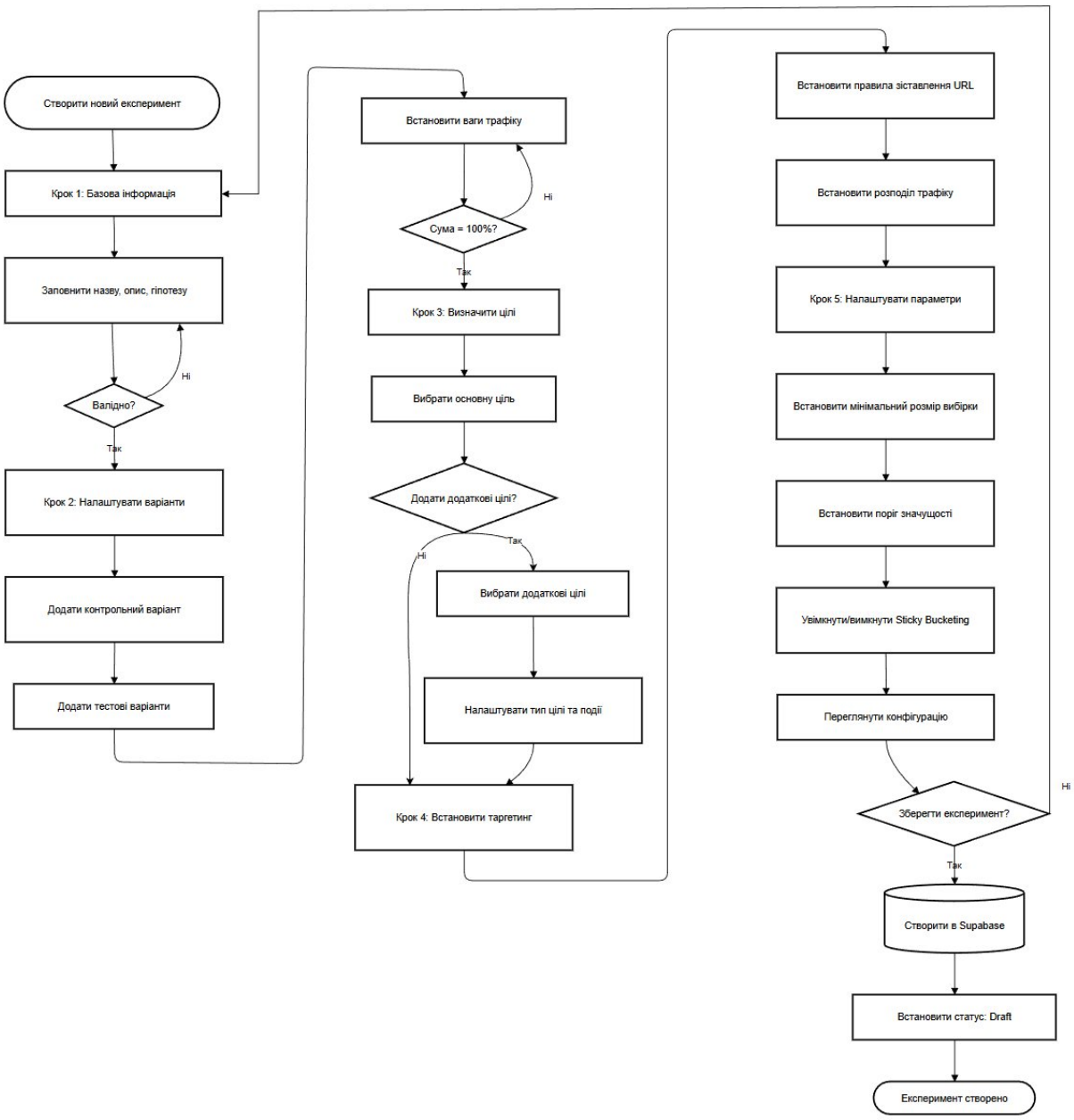


Рисунок А.2 — Процес створення експерименту

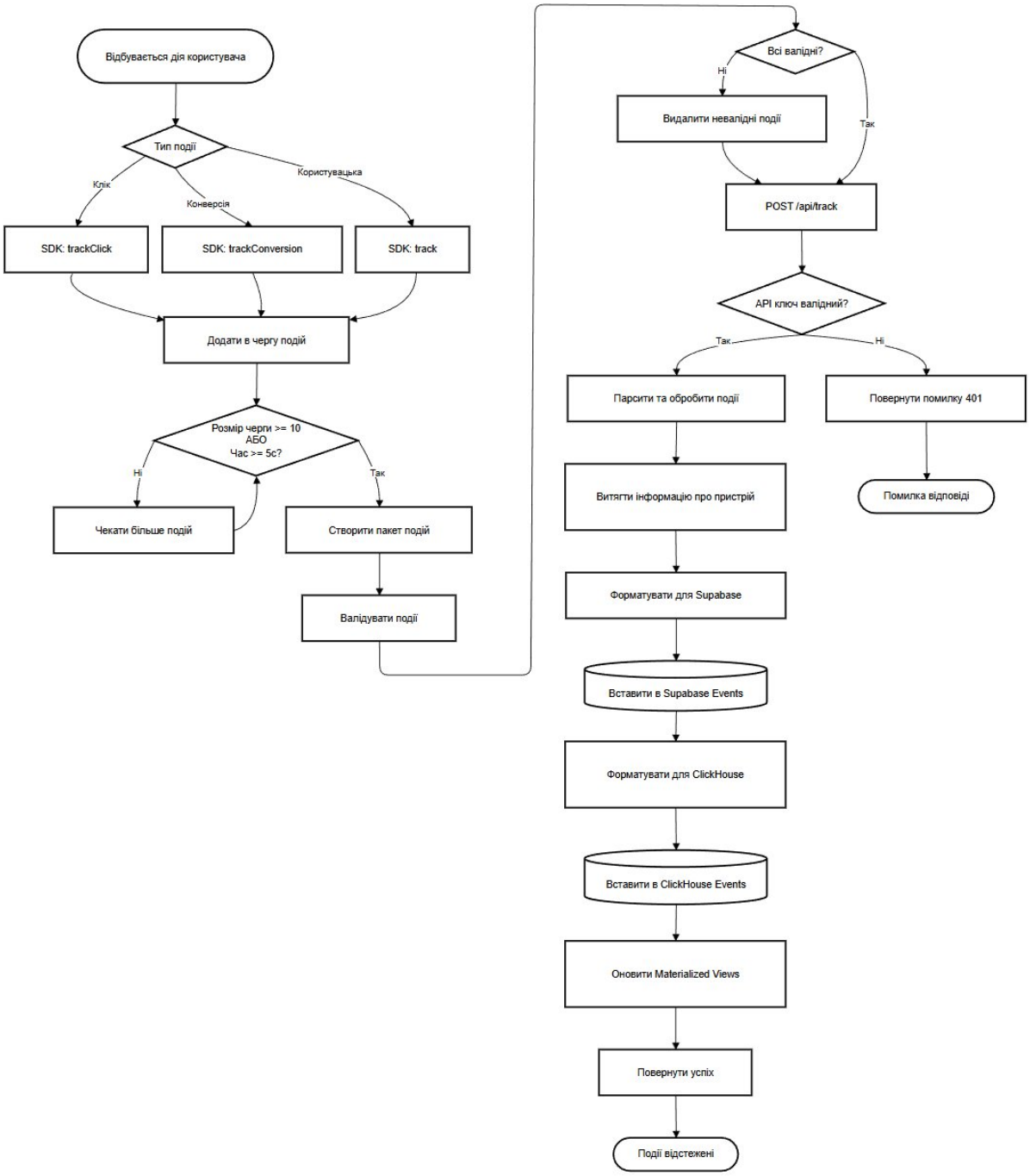


Рисунок А.3 — Потік відстеження подій на клієнтському веб-ресурсі

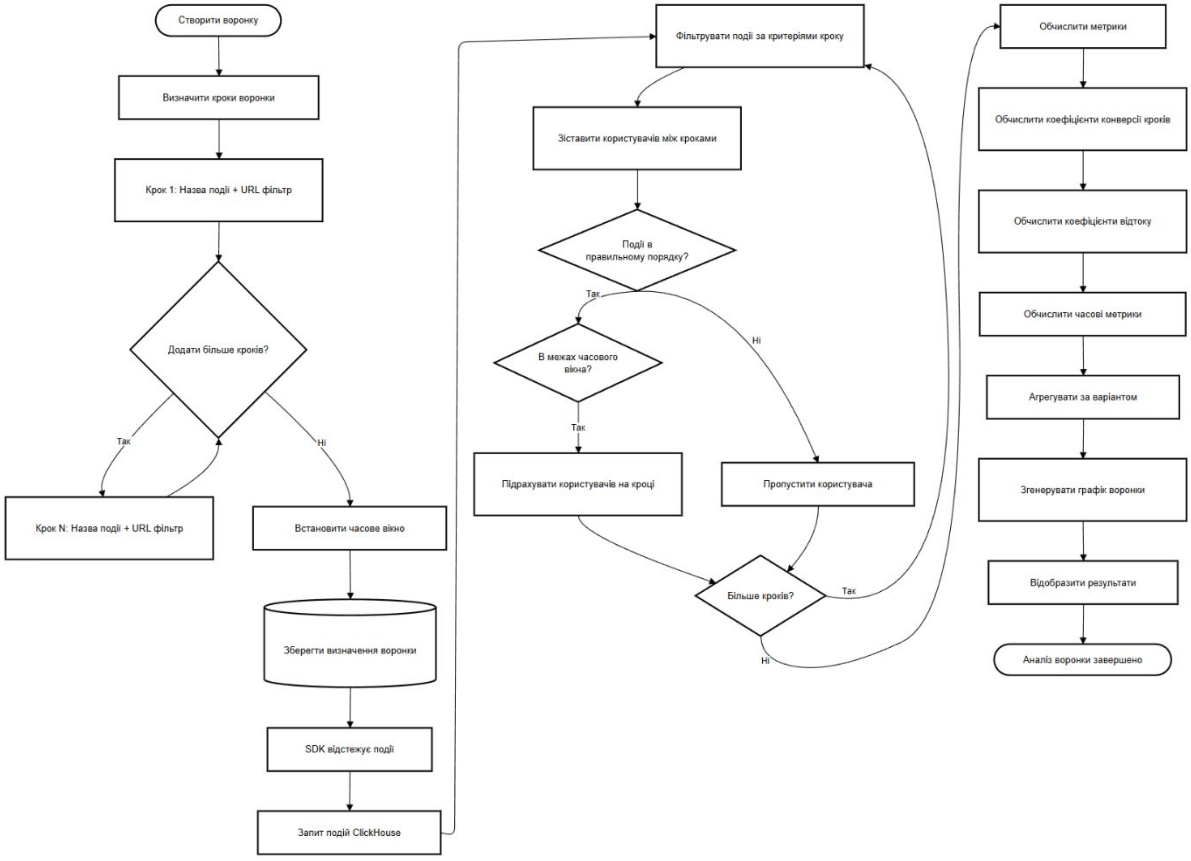


Рисунок А.4 — Процес аналізу воронок конверсії

```
CREATE TABLE public.api_keys (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  project_id uuid NOT NULL,
  name text NOT NULL,
  key text NOT NULL UNIQUE,
  scopes ARRAY DEFAULT ARRAY['track'::text, 'read'::text],
  is_active boolean DEFAULT true,
  last_used_at timestamp with time zone,
  created_at timestamp with time zone DEFAULT now(),
  created_by uuid,
  CONSTRAINT api_keys_pkey PRIMARY KEY (id),
  CONSTRAINT api_keys_project_id_fkey FOREIGN KEY (project_id) REFERENCES public.projects(id),
  CONSTRAINT api_keys_created_by_fkey FOREIGN KEY (created_by) REFERENCES public.users(id)
);
CREATE TABLE public.events (
  event_id uuid NOT NULL DEFAULT gen_random_uuid(),
  session_id text NOT NULL,
  user_id text NOT NULL,
  event_type text NOT NULL,
  event_name text NOT NULL,
  url text,
  referrer text,
  user_agent text,
  properties jsonb DEFAULT '{}'::jsonb,
  experiment_id uuid,
  variant_id uuid,
  timestamp timestamp with time zone DEFAULT now(),
```

```
    created_at timestamp with time zone DEFAULT now(),
    project_id uuid,
    CONSTRAINT events_pkey PRIMARY KEY (event_id),
    CONSTRAINT events_project_id_fkey FOREIGN KEY (project_id) REFERENCES public.projects(id)
);
CREATE TABLE public.experiments (
    id uuid NOT NULL DEFAULT uuid_generate_v4(),
    project_id uuid NOT NULL,
    name text NOT NULL,
    description text,
    hypothesis text,
    status USER-DEFINED DEFAULT 'draft'::experiment_status,
    target_url text,
    url_match_type text DEFAULT 'exact'::text,
    traffic_allocation integer DEFAULT 100 CHECK (traffic_allocation >= 0 AND traffic_allocation <= 100),
    geo_targeting jsonb,
    device_targeting ARRAY,
    browser_targeting ARRAY,
    sticky_bucketing boolean DEFAULT true,
    sample_size_minimum integer,
    significance_threshold numeric DEFAULT 0.95,
    event_tracking jsonb,
    start_date timestamp with time zone,
    end_date timestamp with time zone,
    created_by uuid,
    created_at timestamp with time zone DEFAULT now(),
    updated_at timestamp with time zone DEFAULT now(),
    CONSTRAINT experiments_pkey PRIMARY KEY (id),
```

```
    CONSTRAINT experiments_project_id_fkey FOREIGN KEY (project_id) REFERENCES public.projects(id),
    CONSTRAINT experiments_created_by_fkey FOREIGN KEY (created_by) REFERENCES public.users(id)
);
CREATE TABLE public.funnel_steps (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  funnel_id uuid NOT NULL,
  step_order integer NOT NULL CHECK (step_order >= 0),
  event_name text NOT NULL,
  url_filter text,
  url_match_type text DEFAULT 'contains'::text,
  created_at timestamp with time zone DEFAULT now(),
  CONSTRAINT funnel_steps_pkey PRIMARY KEY (id),
  CONSTRAINT funnel_steps_funnel_id_fkey FOREIGN KEY (funnel_id) REFERENCES public.funnels(id)
);
CREATE TABLE public.funnels (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  project_id uuid NOT NULL,
  name text NOT NULL,
  description text,
  time_window_hours integer DEFAULT 24 CHECK (time_window_hours > 0),
  created_by uuid,
  created_at timestamp with time zone DEFAULT now(),
  updated_at timestamp with time zone DEFAULT now(),
  CONSTRAINT funnels_pkey PRIMARY KEY (id),
  CONSTRAINT funnels_project_id_fkey FOREIGN KEY (project_id) REFERENCES public.projects(id),
```

```
    CONSTRAINT funnels_created_by_fkey FOREIGN KEY (created_by) REFERENCES public.users(id)
);
CREATE TABLE public.goals (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  experiment_id uuid NOT NULL,
  name text NOT NULL,
  description text,
  metric_type USER-DEFINED NOT NULL,
  event_name text,
  is_primary boolean DEFAULT false,
  target_value numeric,
  created_at timestamp with time zone DEFAULT now(),
  CONSTRAINT goals_pkey PRIMARY KEY (id),
  CONSTRAINT goals_experiment_id_fkey FOREIGN KEY (experiment_id) REFERENCES public.experiments(id)
);
CREATE TABLE public.projects (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  name text NOT NULL,
  description text,
  owner_id uuid NOT NULL,
  created_at timestamp with time zone DEFAULT now(),
  updated_at timestamp with time zone DEFAULT now(),
  CONSTRAINT projects_pkey PRIMARY KEY (id),
  CONSTRAINT projects_owner_id_fkey FOREIGN KEY (owner_id) REFERENCES public.users(id)
);
CREATE TABLE public.user_buckets (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
```

```
  experiment_id uuid NOT NULL,
  user_id text NOT NULL,
  variant_id uuid NOT NULL,
  assigned_at timestamp with time zone DEFAULT now(),
  CONSTRAINT user_buckets_pkey PRIMARY KEY (id),
  CONSTRAINT user_buckets_experiment_id_fkey FOREIGN KEY (experiment_id) REFERENCES public.experiments(id),
  CONSTRAINT user_buckets_variant_id_fkey FOREIGN KEY (variant_id) REFERENCES public.variants(id)
);
CREATE TABLE public.users (
  id uuid NOT NULL,
  email text NOT NULL UNIQUE,
  full_name text,
  avatar_url text,
  created_at timestamp with time zone DEFAULT now(),
  updated_at timestamp with time zone DEFAULT now(),
  CONSTRAINT users_pkey PRIMARY KEY (id),
  CONSTRAINT users_id_fkey FOREIGN KEY (id) REFERENCES auth.users(id)
);
CREATE TABLE public.variants (
  id uuid NOT NULL DEFAULT uuid_generate_v4(),
  experiment_id uuid NOT NULL,
  name text NOT NULL,
  description text,
  is_control boolean DEFAULT false,
  traffic_weight integer NOT NULL CHECK (traffic_weight >= 0 AND traffic_weight <= 100),
  config jsonb,
```

```
        created_at timestamp with time zone DEFAULT now(),
        updated_at timestamp with time zone DEFAULT now(),
        CONSTRAINT variants_pkey PRIMARY KEY (id),
        CONSTRAINT variants_experiment_id_fkey FOREIGN KEY (experiment_id)
REFERENCES public.experiments(id)
    );
```