Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

# МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

**«Інформаційна технологія A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей»**

Виконав: студент 2-го курсу, групи 2КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Бугайов В.Ю.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. КН
_____ Козловський А.В.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: д.т.н., професор каф. КСУ
_____ Юхимчук М.С.
(прізвище та ініціали)
«08» 12 2023 р.

**Допущено до захисту**
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«08» 12 2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

**ЗАТВЕРДЖУЮ**
**Завідувач кафедри КН**
**Д.т.н, проф. Яровий А. А.**
**19.08 2023 року**

**З А В Д А Н Н Я**
**НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

Бугайову Володимиру Юрійовичу

1. Тема роботи Інформаційна технологія A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей.
керівник роботи к.т.н., доцент кафедри КН Козловський А. В.
затверджені наказом ВНТУ від «13» 09 2023 р. №247
2. Строк подання студентом роботи 13.11. 2023 року
3. Вихідні дані до роботи:
Операційна система – Windows, Mac OS.
Мова програмування – C#, TypeScript, SQL.
Технології – ASP.NET Core, SQL Server.
4. Зміст текстової частини:

Вступ, аналіз сучасного рівня розвитку інформаційних технологій A/B тестування, моделювання інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей, програмна реалізація інформаційїної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей, економічна частина, висновки, перелік використаних джерел, додатки

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)

Титульний слайд, наукова новизна та практична цінність, огляд платформ для A/B експериментів, загальний алгоритм роботи, архітектура додатку, вікна програми, результати тестування, основні результати роботи, заключний слайд.

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| 1-3 | Козловський А. В., к.т.н., доц. каф. КН | 29.08.23 | 10.11.23 |
| 4 | Ратушняк О. Г., к.т.н., доц. каф. ЕПВМ | 24.10.23 | 1.11.23 |
| | | | |

7. Дата видачі завдання 29.08. 2023 р.

**КАЛЕНДАРНИЙ ПЛАН**

| № з/п | Назва етапів магістерської кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз сучасного рівня розвитку інформаційних технологій A/B тестування | 01.09.2023 – 05.09.2023 | Вик. |
| 2 | Моделювання інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей | 06.09.2023 – 16.09.2023 | Вик. |
| 3 | Програмна реалізація інформаційїної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей | 17.09.2023 – 23.10.2023 | Вик. |
| 4 | Підготовка економічної частини | 24.10.2023 – 01.11.2023 | Вик. |
| 5 | Оформлення пояснювальної записки, графічного матеріалу та презентації | 02.11.2023 – 10.11.2023 | Вик. |
| | | | |

Студент ( підпис) Бугайов В. Ю.

Керівник роботи ( підпис) Козловський А. В..

# АНОТАЦІЯ

УДК 004.42

Бугайов В. Ю. Інформаційна технологія A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей. Магістерська кваліфікаційна робота зі спеціальності 122 «Комп'ютерні науки», освітня програма «Системи штучного інтелекту». Вінниця: ВНТУ, 2023. 103 с.

Укр. мовою. Бібліогр.: 20 назв; рис.: 26; табл. 13.

У магістерській кваліфікаційній роботі «Інформаційна технологія A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей» розроблено програмний додаток, який дозволяє додавати можливість проведеня A/B тестування на вказаному WEB-сайті. У ході роботи отримав подальшого розвитку метод проведення A/B тестування, а саме здійснено поєднання A/B тестування з використанням великих мовних моделей.

В ході виконання роботи було проаналізовано сучасні інформаційні технології, що дають змогу інтегровувати проведення A/B експериментів на WEB-сайтах. Також, було здійснено моделювання інформаційної технології A/B тестування, зпроектовано алгоритм роботи системи. В результатті, було здійснено програмну реалізацію та тестування основної функціональності.

Розроблений продукт має інтуїтивно зрозумілий інтерфейс і є доступний через мережу Інтернет. Це дозволяє мати постійний доступ до всіх даних платформи.

Ключові слова: великі мовні моделі, A/B експерименти, користувацький досвід, WEB-сайти.

# ABSTRACT

Buhaiov V.Y. Information Technology A/B Testing for Website Adaptation Using Large Language Models. Master's thesis in the specialty 122 «Computer science», educational program «Artifical intelligence systems». Vinnytsia: VNTU, 2023. – 103 p.

In Ukrainian language. Bibliogr.: 20 titles; fig. 26; table 13.

The master's qualification work, titled "Information Technology of A/B Testing for Web Adaptation using Large Language Models," has resulted in the development of a software application that enables the implementation of A/B testing on a specified web platform. Throughout the project, the methodology for conducting A/B testing was further enhanced by integrating large language models.

During the execution of this work, a thorough analysis of contemporary information technologies facilitating the integration of A/B testing on websites was conducted. Additionally, the information technology of A/B testing was simulated, and a system workflow algorithm was designed. As a result, the primary functionality was implemented and subjected to testing.

The developed product features an intuitive interface and is accessible over the internet, providing continuous access to all platform data.

Keywords: Large language models, A/B experiments, User experience, WEB sites.

# ЗМІСТ

# ВСТУП

**Актуальність теми дослідження.** Актуальність теми магістерської роботи обумовлена стрімким розвитком інформаційних технологій та потребою у постійному вдосконаленні інтерактивності WEB-сайтів. У цифрову епоху, коли WEB-сайти стають основним обличчям бізнесу в інтернет-просторі, важливість їх адаптації та еволюції є безсумнівною. A/B тестування виступає одним із найпотужніших інструментів для підвищення швидкодії WEB-сайтів, дозволяючи тонко налаштовувати елементи інтерфейсу на основі реальної поведінки користувачів [3]. Це тестування дозволяє вирішувати, які зміни найкраще сприймаються аудиторією, забезпечуючи підвищення конверсійних показників та користувацької взаємодії.

Впровадження штучного інтелекту і, зокрема, великих мовних моделей, у процес A/B тестування дає можливість автоматизувати та оптимізувати цей процес. З використанням AI системи можуть самостійно аналізувати результати тестувань, визначати найбільш ефективні варіанти та навіть пропонувати власні рішення щодо оптимізації контенту. Це відкриває двері до більш персоналізованого та динамічного WEB-дизайну, який може адаптуватися під кожного користувача індивідуально. Мовні моделі, такі як GPT, є перспективними у підвищенні точності та релевантності контенту, адаптованого під потреби користувачів. Інтеграція цих технологій у процес A/B тестування дозволяє не лише тестувати елементи дизайну, але й оптимізувати текстовий контент, що є ключовим для залучення та утримання уваги відвідувачів [4].

Розвиток штучного інтелекту та його застосування в аналітичних процесах значною мірою трансформує методи розробки та оптимізації WEB-сайтів. Використання інтелектуального аналізу даних дозволяє не просто реагувати на зміни у поведінці користувачів, а й прогнозувати ці зміни, підлаштовуючи WEB-ресурси під потенційні інтереси та потреби

відвідувачів. Це відкриває нові горизонти для персоналізації користувацького досвіду, роблячи його більш ефективним і задовільним.

Крім того, внесок мовних моделей у вдосконалення семантичного аналізу та обробки природної мови підсилює здатність WEB-сайтів взаємодіяти з користувачами на більш високому рівні. Вони можуть запропонувати автоматизовані рекомендації, засновані на контексті та попередніх запитах користувача, тим самим підвищуючи ймовірність задоволення їхніх інтересів і потреб. Така інтерактивність, що поєднується із здатністю швидко адаптуватися до змін у користувацьких перевагах, робить WEB-сайти не просто інформаційними ресурсами, а дійсно інтелектуальними помічниками в процесі пошуку та обміну інформацією.

Важливою складовою ефективності веб-сайтів є релевантність їх контенту, яка визначається здатністю контенту задовольняти специфічні потреби та інтереси користувачів. Релевантність можна виміряти через аналіз показників взаємодії користувачів з контентом, таких як час перебування на сторінці, частоту кліків, та ступінь залучення. У цій роботі, релевантність контенту розглядається через використання великих мовних моделей, що дозволяють адаптувати контент до потреб конкретного користувача, забезпечуючи вищу персоналізацію та точність відповідності запитам користувачів.

У сучасному світі, де кожен клік та перегляд є конкурентною перевагою, використання передових технологій для особистісної адаптації WEB-сайтів стає вирішальним фактором успіху.

Розробка нових методик A/B тестування з використанням AI не лише відповідає актуальним потребам ринку, але й вказує на шляхи його майбутнього розвитку.

**Зв’язок роботи з науковими програмами, планами, темами.** Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1 «Розробка прикладних інтелектуальних інформаційних

технологій та систем» та плану наукової та навчально-методичної роботи кафедри.

**Мета та задачі дослідження.** Основною метою дослідження є підвищення швидкодії та релевантності контенту WEB-сайтів за допомогою поєднання процесу A/B тестування і можливостей великих мовних моделей.

**Основними задачами є:**

- Проведення аналізу існуючих програмних засобів для A/B тестування.
- Провести моделювання технології A/B тестування.
- Реалізація програмного забезпечення.
- Оцінка комерційного потенціалу розробки.

**Об’єктом дослідження** є процес взаємодії великих мовних моделей та систем A/B тестування для оптимізації контенту WEB-сайтів.

**Предметом дослідження** є методи інтеграції великих мовних моделей в алгоритми A/B тестування для динамічної оптимізації WEB-контенту.

**Методи дослідження.** У процесі досліджень використовувались: теорія алгоритмів для розробки алгоритму розподілу користувачів між групами; комп’ютерне моделювання для аналізу та перевірки отриманих теоретичних положень, системний аналіз, лінійна алгебра.

**Наукова новизна одержаних результатів.**

– Розроблено удосконалену методику A/B тестування, яка інтегрує можливості штучного інтелекту та великих мовних моделей для оптимізації контенту WEB-сайтів. Методика дозволяє здійснювати більш глибокий семантичний аналіз контенту та його адаптацію до поведінкових факторів користувачів, значно підвищуючи персоналізацію та релевантність WEB-досвіду.

– Запропоновано інформаційну технологію A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей, що відрізняється від існуючих застосуванням інтеграції з великими мовними моделями, що забезпечило підвищення швидкодії та

релевантності контенту WEB-сайтів.

**Практичне значення одержаних результатів** полягає у наступному:

1. Розроблено алгоритм інтеграції A/B тестування з великими мовними моделями.
2. Здійснено програмну реалізацію платформи для проведення A/B тестувань з використанням великих мовних моделей.

**Достовірність теоретичних положень** магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням методів під час доведення наукових положень, порівнянням результатів із відомими, та збіжністю результатів моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

**Особистий внесок автора.** Усі наукові результати, наведені у магістерській кваліфікаційній роботі, отримані автором самостійно. У друкованих працях, опублікованих у співавторстві, автору належать такі результати: аналіз сучасного рівня розвитку інформаційних технологій A/B тестування, методи вирішення задачі адаптації WEB-сайту з використанням A/B тестування та великих мовних моделей.

**Апробація результатів роботи.** Результати досліджень опубліковано у IV Всеукраїнської студентської наукової конференції «Розвиток сучасної науки: актуальні питання теорії та практики» (2023), секція інформаційні технології та системи у м. Львів в листопаді 2023 р., IV Міжнародна наукова конференція «Розвиток наукової думки постіндустріального суспільства: сучасний дискурс», секція інформаційні технології та системи у м. Івано-Франківськ в листопаді 2023 року [1-2].

**Публікації.** За результатами магістерської кваліфікаційної роботи опубліковано дві тези доповідей на науково-технічних конференціях [1-2].

# 1 АНАЛІЗ СУЧАСНОГО РІВНЯ РОЗВИТКУ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ A/B ТЕСТУВАННЯ

## 1.1 Аналіз предметної області A/B тестування для адаптації WEB-сайту

В сучасному цифровому світі, де конкуренція серед WEB-сайтів надзвичайно велика, важливо здійснювати постійне вдосконалення та оптимізацію WEB-сайтів для забезпечення кращого користувацького досвіду та досягнення більшої швидкодії. Одним із найпоширеніших і дієвих інструментів для цього є A/B тестування [4].

A/B тестування – це метод, що використовується для порівняння двох чи більше версій WEB-сайту чи його окремих елементів з метою визначення, яка з них найкраще відповідає вимогам аудиторії та метою сайту. В основі A/B тестування лежить ідея порівняння "версії A" (контрольної) з "версією B" (експериментальною) для виявлення покращень в останній (рис 1.1). Такий підхід дозволяє вирішити важливі питання, такі як оптимізація конверсій, покращення взаємодії з користувачами та збільшення прибутковості WEB-сайту.

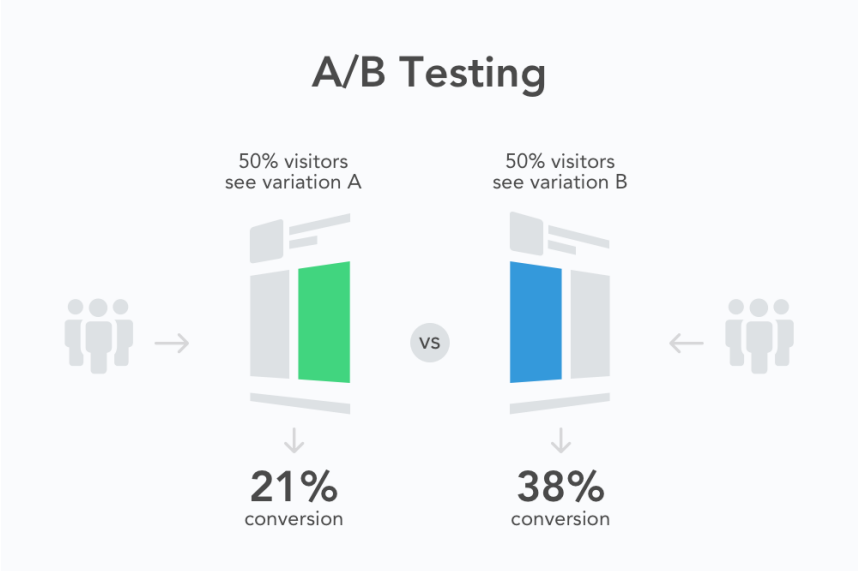



Рисунок 1.1 – Схема A/B тестування

A/B тестування стало необхідним елементом цифрової стратегії WEB-розробників та маркетологів у сучасному Інтернеті [6]. Існують кілька ключових аспектів, які підкреслюють важливість цього методу:

- покращення користувацького досвіду;
- підвищення конверсій;
- мінімізація ризиків;
- підвищення прибутковості.

Покращення користувацького досвіду дозволяє індивідуально визначити та виправити проблеми, які можуть впливати на задоволення користувачів. В результаті, користувачі отримують кращий досвід і, ймовірно, залишаються на сайті довше.

Підвищення конверсій включає в себе той факт, що зміни, які впроваджуються після A/B тестування, часто призводять до збільшення користувачів які користуються продуктом, що означає більше продажів, реєстрацій чи інших бажаних дій користувачів.

Мінімізація ризиків дозволяє зменшити кількість помилок при внесенні змін на WEB-сайті, оскільки вони базуються на даних та результатах експерименту.

Процес A/B тестування складається з декількох кроків, кожен з яких грає важливу роль у забезпеченні надійних та корисних результатів (рис 1.2).

Перший крок у виконанні A/B тестування полягає у визначенні чіткої мети або цілі. Це може бути покращення конверсій, збільшення кількості реєстрацій або зниження кількості відмов. Важливо, щоб ціль була чіткою та вимірюваною.

Після визначення мети, необхідно вибрати змінні, які будуть тестуватися. Це може включати зміни в дизайні WEB-сторінки, текстовому контенті, функціональності чи будь-яких інших аспектах сайту. Далі потрібно розробити дві (або більше) версії WEB-сторінки або її елементів. Одну версію вважають контрольною (версія A), а іншу - експериментальною

(версія B). Версії повинні відрізнятися тільки тими змінними, які ви плануєте перевірити.

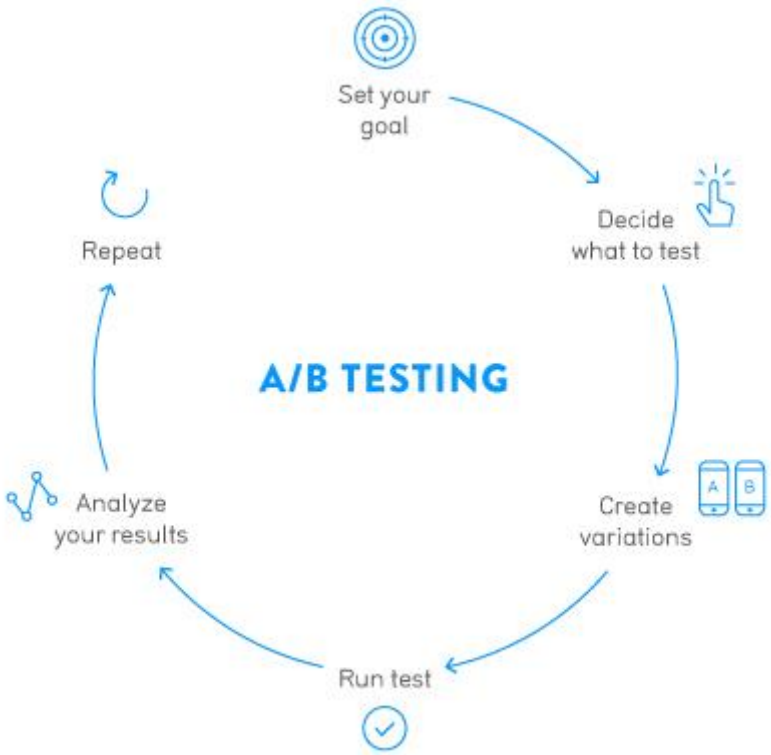



Рисунок 1.2 – Приклад процесу A/B тестування

Тепер розпочинається фаза тестування. Користувачі випадковим чином розподіляються між версією A та версією B, і їх взаємодія зі сторінками вимірюється та записується. Важливо, щоб тестування проводилося протягом достатньо тривалого періоду для збору репрезентативних даних.

Після завершення тестування проводиться аналіз отриманих результатів. Вимірюються показники, які відповідають меті тестування, і порівнюються між версією A та версією B. Зазвичай використовуються статистичні методи для визначення значущості отриманих різниць.

На основі результатів аналізу вирішується, яку версію слід вибрати для подальшого використання на сайті. Зміни, які ввели в експериментальній версії (версія B), можуть бути внесені на основний сайт, щоб покращити його ефективність.

Важливим аспектом сучасного A/B тестування є інтеграція великих мовних моделей для оптимізації WEB-сайтів. Великі мовні моделі, такі як GPT-4, можуть значно підвищити ефективність A/B тестування за рахунок автоматизації процесу аналізу даних та генерації інсайтів. Ці моделі здатні обробляти великі обсяги даних про взаємодію користувачів з сайтом, включаючи кліки, перегляди сторінок та поведінкові патерни. Вони можуть виявляти тонкі шаблони в поведінці користувачів, які можуть бути неочевидними для аналітиків.

Інтеграція великих мовних моделей в процес A/B тестування може сприяти більш точному визначенню контенту, який краще резонує з аудиторією. Також це дає змогу розробникам швидше ідентифікувати та усувати проблемні аспекти на сайті, тим самим покращуючи користувацький досвід. Крім того, використання мовних моделей може допомогти в автоматизації створення варіантів тестування, знижуючи тим самим робочий навантаження на команду розробників.

Особливу увагу слід приділити забезпеченню об'єктивності та точності аналізу даних, що генеруються мовними моделями. Необхідно враховувати, що навіть найсучасніші моделі можуть мати вбудовані упередження або неправильно інтерпретувати контекст, тому важливо поєднувати їх використання з ручним переглядом та аналізом даних. Такий підхід дозволяє досягати високої точності в A/B тестуванні та ефективно адаптувати WEB-сайти під потреби користувачів."

Іншим важливим аспектом застосування великих мовних моделей у контексті A/B тестування є їх здатність генерувати творчі та інноваційні ідеї для експериментальних варіантів WEB-сайту. Мовні моделі можуть автоматично генерувати пропозиції щодо дизайну, контенту та функціональності, які потім можуть бути використані в тестуванні. Це не тільки спрощує процес вибору варіантів для тестування, але й відкриває нові можливості для інновацій та креативних рішень. Використання таких технологій може призвести до значного покращення користувацького

досвіду, збільшення конверсій та підвищення загальної ефективності WEB-сайту. Тим не менш, важливо підходити до інтеграції цих інновацій з обережністю, враховуючи специфіку цільової аудиторії та цілісність бренду, щоб забезпечити їх ефективну та збалансовану впровадження.

A/B тестування має багато переваг, включаючи науковий підхід до оптимізації WEB-сайтів, здатність до зменшення ризиків при внесенні змін та покращення користувацького досвіду. Однак важливо також розуміти обмеження цього підходу, включаючи обмеження в ресурсах для проведення тестів та можливість отримати хибно-позитивні результати.

## 1.2 Аналіз методів та засобів вирішення задачі адаптації WEB-сайту з використанням A/B тестування та великих мовних моделей

A/B тестування включає в себе багато підходів, існують численні інструменти і методи, які можна використовувати для вирішення задачі адаптації WEB-сайту, тому варто розглянути деякі з них.

Однією з ключових можливостей використання великих мовних моделей в A/B тестуванні є здатність до автоматизації процесу збору та аналізу зворотного зв'язку від користувачів. Це може включати аналіз коментарів, відгуків, а також вивчення моделей поведінки користувачів на сайті. Великі мовні моделі можуть обробляти масивні обсяги даних, виявляючи неочевидні взаємозв'язки та впливи, які можуть бути важливими для прийняття рішень щодо адаптації WEB-сайту. Завдяки цьому, вони сприяють розробці більш точних та ефективних стратегій для оптимізації веб-інтерфейсів, дозволяючи враховувати багатогранність користувацьких переваг та очікувань.

Візуальне A/B тестування дозволяє проводити порівняння між різними версіями WEB-сайту, фокусуючись на зовнішньому вигляді та дизайні. Цей метод корисний для визначення того, яка графічна версія привертає більше

уваги та сприяє більшому взаємодії користувачів. Він може включати тестування різних кольорових схем, шрифтів, розташування елементів на сторінці та інших аспектів WEB-дизайну.

Текстове A/B тестування концентрується на аналізі текстового контенту WEB-сайту, такого як заголовки, описи продуктів, кнопки виклику до дії та інші текстові елементи. Цей метод допомагає визначити, які текстові варіанти привертають більше уваги користувачів і підвищують конверсію.

Функціональне A/B тестування оцінює різні аспекти функціональності WEB-сайту. Це включає в себе тести на визначення ефективності нових функцій, покращень користувацького інтерфейсу, способів навігації та інших елементів, які можуть вплинути на користувацький досвід.

Мультиваріантне тестування – це розширена версія A/B тестування, де тестується більше двох варіантів. Цей метод дозволяє вивчити вплив багатьох різних змінних одночасно. Він особливо корисний у випадках, коли вам потрібно оцінити вплив багатьох аспектів сайту на одній сторінці.

Сегментація та персоналізація дозволяють створити різні версії WEB-сайту для різних груп користувачів. Це дозволяє адаптувати сайт до потреб конкретних аудиторій. Наприклад, ви можете створити спеціальну версію сайту для нових користувачів та іншу версію для постійних клієнтів.

A/B/C тестування розширює концепцію A/B тестування, дозволяючи одночасно порівнювати три різні варіанти (версія A, версія B і версія C). Це корисний метод, коли вам потрібно визначити найкращий із трьох варіантів.

Кожен з цих методів має свої переваги і обмеження, і вибір методу залежить від конкретної завдання та мети A/B тестування. Іноді комбінація декількох методів може бути найкращим рішенням для досягнення бажаних результатів. В розроблюваній інформаційні технології буде використовуватися комбінація методів візуального та текстового A/B тестування.

Великі мовні моделі (LLM) – це потужні інструменти для аналізу та обробки текстової інформації (рис 1.3). Вони базуються на глибокому

навчанні та штучних нейронних мережах і володіють вражаючою здатністю розуміти та генерувати текст [7]. Використання великих мовних моделей в контексті A/B тестування може значно полегшити процес аналізу результатів та прийняття рішень.

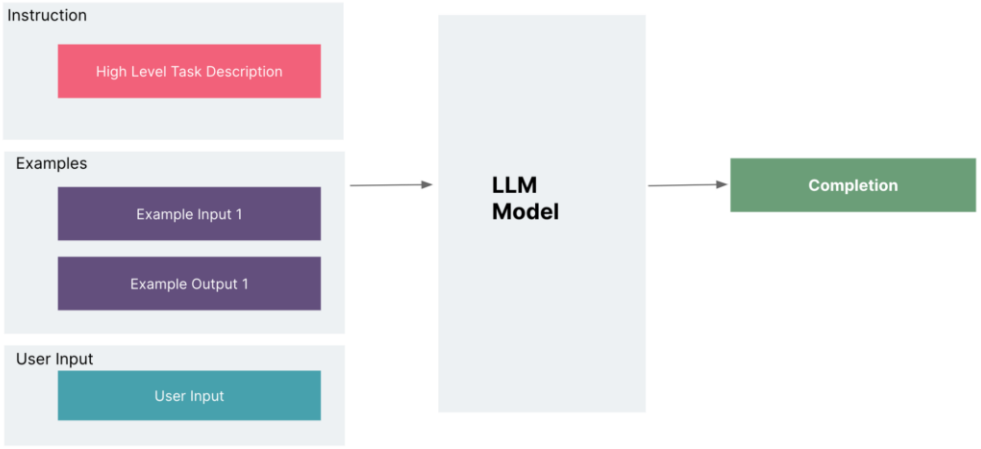



Рисунок 1.3 – Приклад роботи великої мовної моделі

Важливою перевагою великих мовних моделей є їхня здатність аналізувати текст на рівні семантики та контексту. Вони можуть виявляти патерни, які можуть залишитися непоміченими іншими методами. Наприклад, за допомогою LLM можна аналізувати коментарі користувачів, визначати їхні погляди та відчуття щодо змін на WEB-сайті, а також прогнозувати можливі реакції.

Під час A/B тестування великі мовні моделі можуть бути використані для аналізу текстового та контентного вмісту WEB-сайту, виявлення впливу змін на користувачів та формування рекомендацій щодо оптимізації. Вони допомагають розуміти, які текстові та семантичні аспекти важливі для користувачів, і допомагають вибирати найбільш ефективні стратегії адаптації WEB-сайту.

Великі мовні моделі можуть бути використані не лише для аналізу тексту, але і для автоматизованої генерації текстового контенту, такого як заголовки, описи, рекламні повідомлення тощо. Це робить їх універсальними інструментами для покращення користувацького досвіду на WEB-сайтах.

Зазначена універсальність великих мовних моделей робить їх незамінними для підвищення ефективності маркетингових кампаній та оптимізації контенту. Наприклад, вони можуть створювати персоналізовані рекламні повідомлення, які враховують індивідуальні інтереси користувачів, підвищуючи ймовірність їх взаємодії з рекламою.

Використання великих мовних моделей у сфері A/B тестування дозволяє знизити витрати часу та ресурсів, які зазвичай потрібні для аналізу великих обсягів текстової інформації. Вони можуть автоматизувати процес визначення найкращих підходів та забезпечити швидше прийняття рішень щодо оптимізації WEB-сайту. Все це робить їх потужними інструментами для покращення користувацького досвіду та досягнення бажаних цілей в області WEB-розробки та маркетингу.

У даній розробці великі мовні моделі будуть використовуватися для генерації контенту під час виконання A/B експерименту. Також, буде реалізована можливість можливість використання різних провайдерів великих мовних моделей.

## 1.3 Аналіз існуючих програмних засобів проведення A/B тестування

Аналіз існуючих програмних засобів для проведення A/B тестування – є важливим етапом при розробці даної інформаційної системи. Вибір правильного інструменту кінцевим користувачем може суттєво вплинути на успішність та ефективність експерименту. У цьому розділі ми розглянемо деякі з наявних програмних засобів для проведення A/B тестування.

Google Optimize є одним із популярних інструментів для виконання A/B тестування та персоналізації WEB-сайтів [8]. Він пропонує безкоштовну версію з можливістю розширення функціоналу за допомогою платних планів. Google Optimize дозволяє легко створювати та виконувати тести, використовуючи власний візуальний редактор, і аналізувати результати на

основі різних метрик. Важливо відзначити, що цей інструмент інтегрується з іншими Google-продуктами, такими як Google Analytics, що робить його зручним для користувачів, які вже використовують ці інструменти.

VWO (Visual Website Optimizer) – це інструмент для візуального A/B тестування, який надає можливість легко створювати та виконувати тести на WEB-сайті. Він спрощує процес створення різних варіацій сторінок та взаємодії з ними, не вимагаючи глибоких знань програмування. VWO також пропонує інші функції, такі як аналіз теплових карт і персоналізація вмісту для користувачів.

Adobe Target – інструмент від компанії Adobe, який пропонує рішення для A/B тестування, персоналізації та мультитестування [9]. Він інтегрується з іншими продуктами Adobe, що дозволяє користувачам легко керувати та аналізувати результати експериментів. Adobe Target також підтримує таргетування на основі різних параметрів, що дозволяє створювати персоналізований контент для користувачів (рис 1.4).

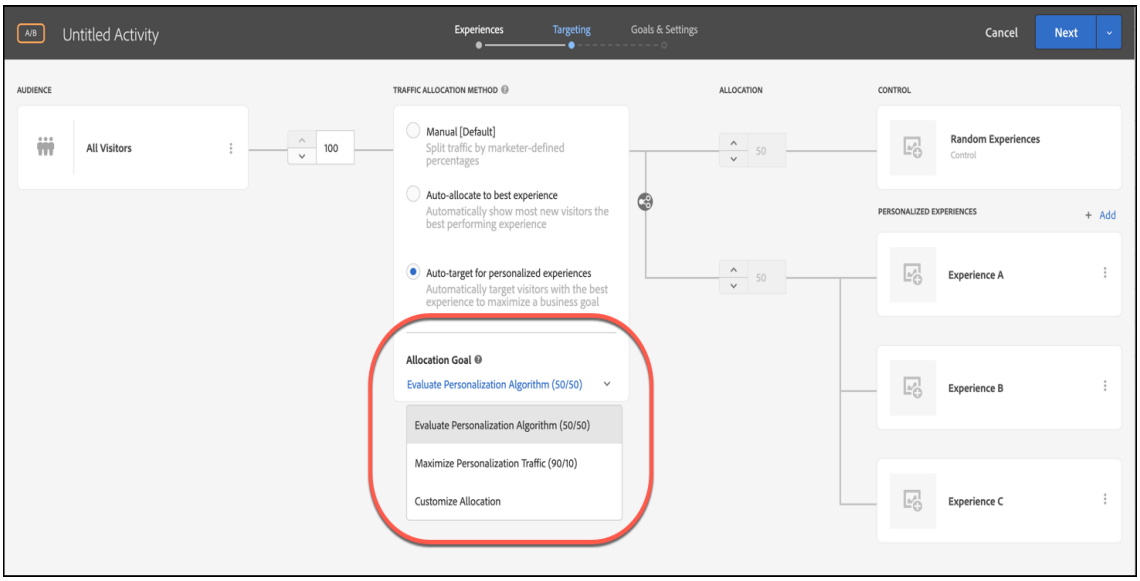



Рисунок 1.4 – Приклад налаштування A/B експерименту в Adobe Target

Crazy Egg – це інструмент, який спеціалізується на аналізі користувацької активності на WEB-сайті [10]. Він дозволяє створювати теплові карти, записи відвідувань та проводити A/B тестування, щоб

визначити, які зміни на сторінці можуть покращити користувацький досвід. Після встановлення на WEB-сайт, Crazy Egg збирає докладну інформацію про споживачів, які заходять на сайт, допомагаючи користувачам зрозуміти поведінку відвідувачів та приймати обґрунтовані рішення щодо оптимізації.

Crazy Egg також надає інтуїтивно зрозумілі звіти та візуалізації, які дозволяють користувачам легко інтерпретувати зібрані дані. Звіти включають і скрол-карти, які показують, наскільки далеко користувачі прокручують сторінки, та клік-карти, які відображають, на які елементи сторінки найчастіше натискають. Це є цінним для виявлення ключових зон інтересу на сторінці та оптимізації її структури та дизайну.

Проаналізувавши усі аналоги, визначено їхні можливості та недоліки, які враховувались при створенні власного програмного забезпечення з назвою «A/B testing with LLM» (табл. 1.1).

Таблиця 1.1 – Порівняльна характеристика аналогів

| Критерій | Google Optimize | Adobe Target | Crazy Egg | A/B testing With LLM |
|---|---|---|---|---|
| Підтримка декількох варіацій | Так | Так | Так | Так |

Продовження табл 1.1

| | | | | |
|---|---|---|---|---|
| Можливість самостійного розгортання | Ні | Ні | Ні | Так |
| Вбудована аналітика | Так | Ні | Так | Так |
| Можливість виконання декількох тестів | Так | Ні | Так | Так |
| Підтримка великих мовних моделей | Ні | Ні | Ні | Так |

Аналіз порівняльних характеристик вказує на доцільність розробки програмного продукту. Як результат, розроблений продукт вирішуватиме недоліки існуючих рішень і забезпечуватиме розширений набір функцій, які необхідні для проведення A/B тестування і відрізнятиметься відмінною від інших підтримкою роботи з різними постачальниками великих мовних моделей та можливістю самостійного розгортання системи.

## 1.4 Постановка задачі реалізації інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей

Для успішної реалізації інформаційної технології A/B тестування з використанням великих мовних моделей важливо чітко сформулювати завдання, які слід вирішити. Це включає в себе не тільки вибір і аналіз інструментів та методів тестування, але й розуміння потреб кінцевих користувачів та ринкового потенціалу розробки. Специфічність використання великих мовних моделей у процесі A/B тестування вимагає глибокого аналізу їх можливостей та обмежень, щоб забезпечити максимальну ефективність і точність результатів тестування. Постановка цих завдань визначає наступні кроки у розробці і використанні цієї технології. Завдання включають в себе наступне:

1. Проведення аналізу існуючих програмних засобів для A/B тестування.
    1.1 Провести огляд існуючих програмних реалізацій A/B тестування.
    1.2 Визначити переваги та недоліки існуючих рішень.
2. Провести моделювання технології A/B тестування.
    2.1 Визначити необхідні компоненти та модулі для системи.
    2.2 Розробити структурну схему компонентів.
    2.3 Запроектувати базу даних

2.4 Описати використання великих мовних моделей.

2.5 Розробка алгоритму роботи системи.

3. Реалізація програмного забезпечення.

3.1 Вибрати оптимальні засоби та платформу для розробки.

3.2 Реалізувати вихідний код компонентів.

3.3 Провести тестування та валідацію програмного забезпечення.

4. Оцінка комерційного потенціалу розробки.

4.1 Прогнозування витрат на науково-дослідну, дослідно-конструкторську та конструкторсько-технологічну роботу.

4.2 Розрахунок ефективності інвестицій та періоду їх окупності.

Визначивши задачі розробки, можливо розпочинати визначення та аналіз методів й засобів, моделювання й реалізацію технології A/B тестування з використанням великих мовних моделей та опис відповідних технічних та економічних показників, що відносяться до розробки.

## 1.4 Висновок до розділу 1

У першому розділі наголошено на важливості та актуальності створення інформаційної системи A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей. Проведено аналіз сучасних методів проведення A/B тестів та розглянуто існуючі системи для адаптації WEB-сайтів за допомогою A/B тестування. Результати порівняння відображають доцільність розробки власного програмного рішення для досягнення поставлених цілей.

Визначено основні завдання, які необхідно виконати під час реалізації інформаційної технології.

# 2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ A/B ТЕСТУВАННЯ ДЛЯ АДАПТАЦІЇ WEB-САЙТУ З ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

## 2.1 Аналіз та обґрунтування вибору складових компонентів інформаційної технології

Під час розробки інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей, важливим етапом є аналіз та обґрунтування вибору складових компонентів інформаційної технології. Перед розробкою програмного забезпечення, необхідно визначити, які компоненти входитимуть в склад технології та як вони будуть взаємодіяти.

Інформаційна технологія (ІТ) в даному контексті представляє собою сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів. ІТ грає ключову роль у розробці системи A/B тестування, оскільки вона забезпечує збереження та обробку даних, які використовуються під час тестування та аналізу результатів.

Основною інформаційної технології для системи A/B тестування буде сервіс, що буде відповідати за повернення даних для виконання експерименту кінцевим користувачам, і також буде відповідати за налаштування експериментів, що буде здійснюватися через клієнтську частину. Сервіс повинен мати чітко визначені виклики, які можуть здійснюватися до нього, для уникнення порушення контрактів між ним і клієнтом та в загальному помилок.

Важливою складовою системи також є панель управління експериментами. За допомогою неї можливо створити новий експеримент, задати умови попадання на експеримент, також створити нові варіації та вказати код, який буде виконуватися при попаданні на них.

Ще однією компонентою системи є код, що виконується коли користувач відвідує сайт. Основна задача даної функціональності на основі активних експериментів зробити перевірити чи користувач потрапляє під умову, що вказана в експерименті, і зробити розподіл в яку групу поточний користувач має потрапити. Ділення трафіку на різні варіації є важливою складовою при проведенні A/B тестування [11]. Розподіл трафіку на варіації вимагає певного розуміння та методології для забезпечення надійності результатів. Одним із поширених методів розподілу трафіку є розділення його на основі відсотків.

Щоб випадково визначити відсоткову варіацію в межах від 0 до 100, можна використовувати генератор випадкових чисел. У багатьох програмних мовах та середовищах розробки є вбудовані функції чи бібліотеки для генерації випадкових чисел. Приклад того, як це можна зробити на мові програмування JavaScript зображено на рисунку 2.1.

```
// Генеруємо випадкове число від 0 до 100
const randomPercentage = Math.floor(Math.random() * 101);

// Виводимо результат
console.log(`Випадковий відсоток: ${randomPercentage}`);
```

Рисунок 2.1 – Генерування випадкових чисел

У цьому прикладі ми використовуємо Math.random(), який генерує випадкове число в межах від 0 (включно) до 1 (виключно). Потім ми множимо це число на 100 і використовуємо Math.floor(), щоб отримати ціле число від 0 до 100. Знайдене відсоткове значення можна використовувати для розподілу трафіку на різні варіації відповідно до задачі експерименту.

## 2.2 Побудова структурної схеми інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей

Структурна схема визначає структуру та компоненти системи, які спрямовані на покращення користувацького досвіду та оптимізацію WEB-сайту в реальному часі. Правильний вибір компонентів визначає ефективність та функціональність технології [12].

На рисунку 2.2 наведена структурна схема компонентів розроблюваної інформаційної технології A/B тестування.

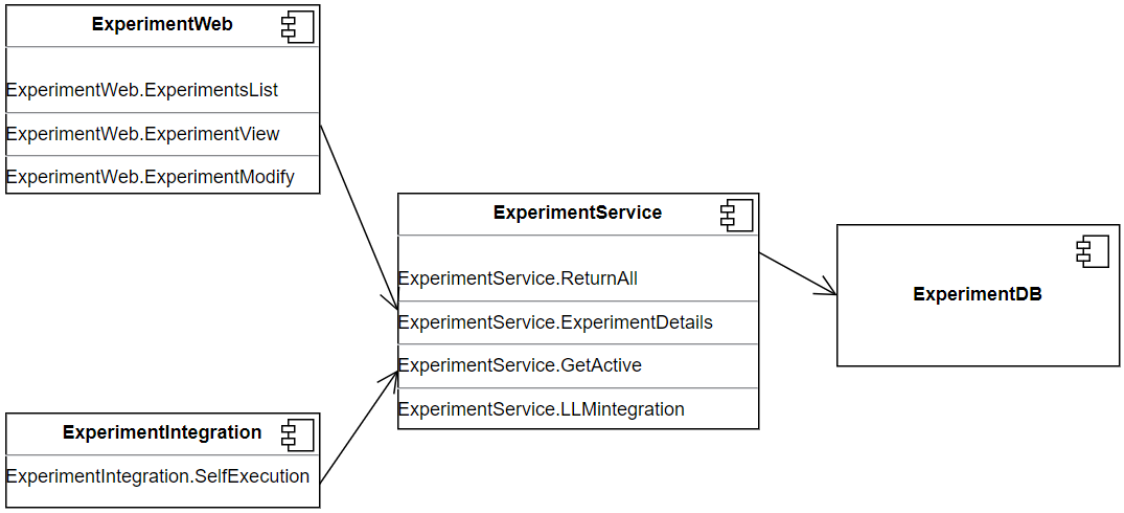



Рисунок 2.2 – Структурна схема інформаційної технології

В побудованій схемі кожен компонент представляє собою окремий структурний блок, а його назва вказана у верхній частині. Далі розглянемо призначення та функціональність кожного компонента:

1. ExperimentWeb – компонент клієнтської частини інформаційної технології, він є панеллю управління експериментами.
   1.1 ExperimentWeb.ExperimentsList – модуль клієнтської частини, що відповідає за витягування та відображення списку всіх експериментів.

1.2 ExperimentWeb.ExperimentView – даний модуль відповідає за детальне відображення інформації про кожен окремий експеримент.

1.3 ExperimentWeb.ExperimentModify – модуль відповідає за налаштування експериментів та створення нових варіацій для експерименту.

2. ExperimentService – це компонент серверної частини, що відповідає за обробку запитів від клієнтської частини та від кінцевих користувачів, що користуються WEB-сайтом, на якому впроваджена система.

2.1 ExperimentService.ReturnAll – модуль серверної частини, що повертає список всіх експериментів, доступний лише для авторизованих користувачів.

2.2 ExperimentService.ExperimentDetails – повертає деталі про експеримент, ідентифікатор, якого було передано.

2.3 ExperimentService.GetActive – повертає список активних експериментів та деталі про них, для подальшого виконання в браузері кінцевого користувача сайту.

2.4 ExperimentService.LLMintegration – модуль відповідає за роботу з різними провайдерами великих мовних моделей.

3. ExperimentDB – компонент, що відповідає за базу даних, в які буде зберігатися інформація про експерименти, їх стан та аналітика.

4. ExperimentIntegration – компонент, що відповідає за виконання коду експериментів.

4.1 ExperimentIntegration.SelfExecution – модуль за визначення потрібної варіації для експерименту та виконання коду цієї варіації.

Ця структурна схема визначає взаємозв'язки між ключовими компонентами інформаційної технології A/B тестування з використанням

великих мовних моделей для адаптації WEB-сайту. Використання великих мовних моделей стає важливим інструментом для поліпшення користувацького досвіду та досягнення ефективності сайту в сучасному цифровому середовищі.

## 2.3 Інтеграція з великими мовними моделями для підтримки адаптації WEB-сайту

Великі мовні моделі володіють унікальною здатністю створювати високоякісний та цільовий контент. Інтеграція LLM дозволить автоматично генерувати та оптимізувати текст для сайту, покращуючи користувацький досвід та забезпечуючи належну адаптацію для кожного користувача. Великі мовні моделі можуть автоматично адаптувати контент до індивідуальних потреб кожного користувача, враховуючи їхні вподобання та контекст використання. Це допомагає створити більш персоналізований контент.

Використання великих мовних моделей у процесі A/B тестування дає можливість генерувати різні варіації контенту для тестів, що допомагає визначити, які зміни є найефективнішими для досягнення встановлених цілей. Ця інтеграція робить A/B тестування більш точним та результативним.

Автоматизація процесу генерації тексту спрощує завдання контент-менеджерів та розробників. Вони можуть зосередитися на стратегічних завданнях, а інтелектуальна система забезпечить необхідний контент.

В розроблюваній технології запланована можливість гнучкого переключення між використанням різних провайдерів великих мовних моделей. Завдяки гнучкій інтеграції, можливо дозволити користувачам обирати та змінювати провайдера з легкістю, без необхідності внесення значних змін у систему. Різні провайдери LLM можуть володіти різними ресурсами та здатностями.

Роблячи нашу інтеграцію гнучкою, ми зменшуємо ризик та залежність від одного провайдера. Якщо виникне необхідність перейти до іншого

провайдера або використовувати кілька провайдерів одночасно, це буде зроблено без серйозних труднощів.

Використання великих мовних моделей є сучасною практикою, яка надає значні переваги серед інших аналогів.

У JavaScript коді для того, щоб відбувся виклик сервісу інтеграції великих мовних моделей, можна використовувати плейсхолдери для тексту питання. Ці плейсхолдери будуть встановлюватися в коді відповідного експерименту. Приклад, як це може виглядати наведено на рисунку 2.3.

```
// Початковий код для виклику сервісу LLM
const question = "{{Whether today in Vinnytsia}}"; // Плейсхолдер для тексту питання

// Виклик сервісу LLM з текстом питання
const response = callLLMService(question);

// Отримана відповідь від LLM
console.log("Відповідь LLM:", response);
```

Рисунок 2.3 – Використання плейсхолдерів

У даному коді "{{Whether today in Vinnytsia}}" є плейсхолдером для тексту питання. При запуску конкретного експерименту, можливо замінити цей плейсхолдер будь-яким іншим текстом питання, що відповідає експерименту. Наприклад, якщо експеримент передбачає перекладання тексту, тоді генерацію тексту для запиту LLM можна замінити на питання з проханням перекласти конкретний текст на різні мови в залежності від певних умов.

Цей підхід дозволяє динамічно змінювати текст питання в коді для кожного експерименту, зберігаючи структуру скрипту, і легко забезпечувати взаємодію з LLM сервісом для кожного конкретного завдання.

## 2.4 Проектування бази даних

Під час розробки інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей, виникає необхідність проектування бази даних. Проектування бази даних - це важливий етап, де створюється схема бази даних та визначаються обмеження цілісності [13].

В рамках цієї технології, проектування бази даних включає в себе такі завдання:

1. Забезпечення зберігання у базі даних всієї необхідної інформації, пов'язаної з експериментами та адаптацією WEB-сайту з використанням великих мовних моделей.

2. Забезпечення можливості отримання даних з бази даних за допомогою різних запитів, що включають в себе велику кількість параметрів для налаштування експериментів.

3. Мінімізація надмірності і дублювання даних шляхом оптимізованої організації даних в базі даних.

4. Забезпечення цілісності бази даних, а також захист від несанкціонованого доступу та втрати даних.

У контексті цієї технології, база даних повинна містити інформацію про всі створені експерименти, їх поточний стан (наприклад, чи вони активні чи призупинені), варіації експериментів та код, який буде використовуватися для модифікації сторінок WEB-сайту.

Розробка універсальних таблиць для бази даних є важливою складовою проектування. Такі таблиці включають в себе інформацію про всі атрибути даних, які використовуються в рамках даної технології [14]. У великих проектах такі універсальні таблиці можуть бути подальше декомпозовані для оптимізації та підтримки вимог проекту. База даних додатку має містити атрибути, представлені в таблиці 2.1, яка описує сутності та зв'язки, важливі для інформаційної технології A/B тестування та адаптації WEB-сайту.

Таблиця 2.1 – Перелік атрибутів інформаційної системи

| № | Назва атрибута | Ім’я поля | Коментар |
|---|---|---|---|
| 1 | ID моделі | ModelId | Ідентифікатор моделі мовної обробки |
| 2 | Назва моделі | ModelName | Назва великої мовної моделі |
| 3 | Параметр моделі | ModelParameters | Параметри конфігурації моделі |
| 4 | Тип мовної моделі | ModelType | Тип великої мовної моделі |
| 5 | Ідентифікатор експерименту | Expeirment_Id | Ідентифікатор експерименту |
| 6 | ID експерименту | ExperimentId | ID експерименту |
| 7 | Назва експерименту | ExperimentName | Назва експерименту |
| 8 | Дата створення | ExperimentDate | Дата створення експерименту |
| 9 | Код для виконання на сторінці | ExecutionCode | Код для виконання на сторінці WEB-сайту |
| 10 | Відсоток трафіку на варіації | PercentageOfTraffic | Відсоток користувачів, що потраплять на варіацію |
| 11 | Ідентифікатор користувача | User_Id | Унікальний ідентифікатор користувача |
| 12 | Тип користувача | User_Type | Тип користувача |

На основі таблиці можна виділити сутності та їх атрибути для інформаційної технології:

1. Модель (Model);

1.1 ID моделі (ModelId): Унікальний ідентифікатор моделі мовної обробки;

1.2 Назва моделі (ModelName): Назва великої мовної моделі;

1.3 Параметри моделі (ModelParameters): Параметри конфігурації моделі;

1.4 Тип мовної моделі (ModelType): Тип великої мовної моделі;

2. Експеримент (Experiment);

2.1 Ідентифікатор моделі (ModelId): Ідентифікатор використаної моделі мовної обробки в експерименті;

2.2 ID експерименту (ExperimentId): Унікальний ідентифікатор експерименту;

2.3 Назва експерименту (ExperimentName): Назва експерименту;

2.4 Дата створення (ExperimentDate): Дата створення експерименту;

2.5 Код для виконання на сторінці (ExecutionCode): Код для виконання на сторінці WEB-сайту;

2.6 Відсоток трафіку на варіації (PercentageOfTraffic): Відсоток користувачів, що потраплять на варіацію;

3. Користувач (User);

3.1 Ідентифікатор експерименту (Expeirment_Id): Ідентифікатор експерименту, з яким користувач асоційований;

3.2 ID користувача (User_Id): Унікальний ідентифікатор користувача;

3.3 Тип користувача (User_Type): Тип користувача.

Ця структура атрибутів відображає основні об'єкти та взаємозв'язки між ними, необхідні для впровадження інформаційної технології A/B тестування з використанням великих мовних моделей на WEB-сайті.

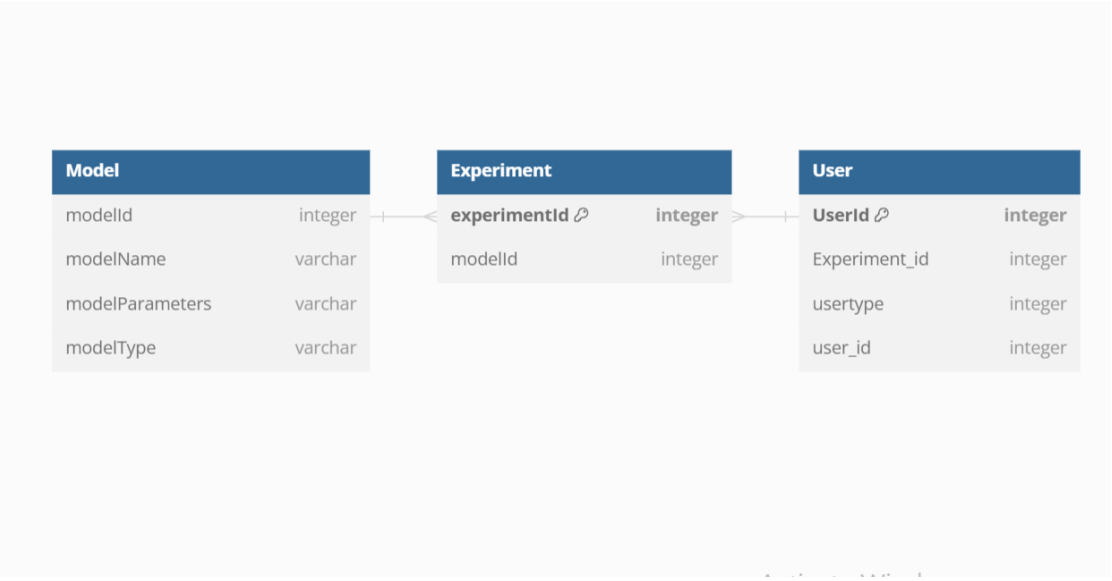



Рисунок 2.4 – Структура бази даних

На основі вищеописаних сутностей була створена база даних, структура якої наведена на рисунку 2.4.

## 2.5 Розробка алгоритму роботи інформаційної системи

Етап проектування алгоритмів включає розробку логіки кожного алгоритму та визначення послідовності операцій, необхідних для досягнення бажаних результатів [15]. При проектуванні алгоритмів наголошується на визначенні послідовності операцій, необхідних для досягнення бажаних результатів. Наприклад, алгоритм взаємодії із великими мовними моделями може включати наступні кроки:

1. Введення текстового запиту від користувача для варіації
2. Виклик великої мовної моделі для обробки запиту.
3. Аналіз результату та вибір оптимальної відповіді.
4. Повернення відповіді користувачеві.

Важливим алгоритмом системи, є загальний алгоритм роботи. Оскільки система є досить складна, важливість швидкого оброблення запитів та отримання результатів є важливим фактором в проектуванні алгоритму.

Алгоритм складається із 10 кроків. Вхідними даними для алгоритму є наявність доступу в користувача до панелі управління експериментами. Це дасть змогу користувачу створити новий експеримент. Для створення експерименту користувачу потрібно мати інформацію про умови потрапляння на експеримент, назву експерименту, скільки варіацій буде, та які зміни вони будуть містити в собі. Якщо це буде експеримент, що буде викликати великі мовні моделі, необхідно тоді підготувати заздалегідь сформовані питання, що будуть добавлені в код. Після внесення всіх даних до панелі управління експериментами, необхідно їх зберегти. Наступним кроком залишається запуск експерименту, та тестування його роботи.

Схема алгоритму зображена на рисунку 2.5.

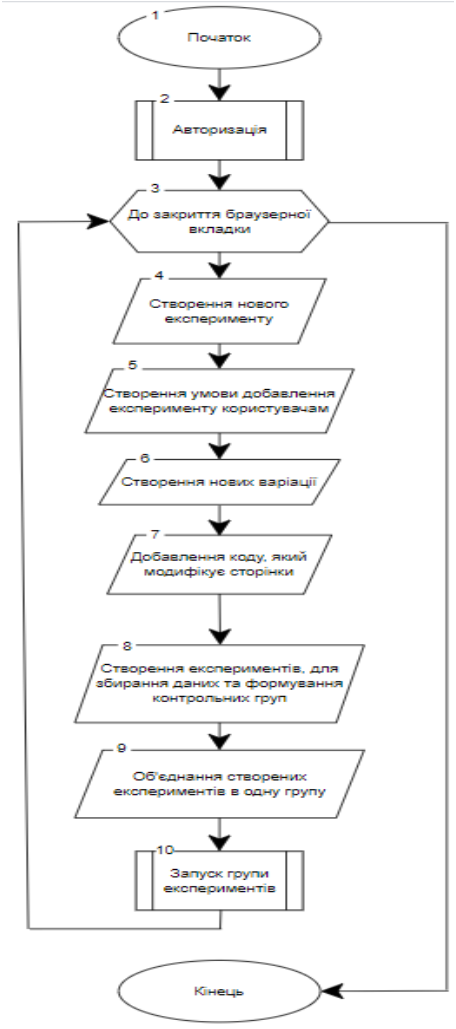



Рисунок 2.5 – Алгоритм роботи програми

Також алгоритм містить крок об’єднання декількох експериментів в одну групу. Це дає змогу запускати декілька різних експериментів одночасно для можливості перевірки комбінації різних змінних одна з одною. Результат запуску експерименту слугує вихідними даними з алгоритму.

Отже, розроблено алгоритм роботи програми, котрий відображає гнучкість системи по відношенню до роботи з A/B експериментами та використанням великих мовних моделей.

## 2.6 Висновок до розділу 2

У другому розділі було проведено докладний аналіз основних компонентів розроблюваної системи. Описано основні особливості та побудовано структурну схему системи, що дозволяє чітко визначити взаємозв’язки та взаємодію між компонентами.

Обґруновано інтеграцію з великим мовними моделями. Ця інтеграція відкриває можливості у сфері аналізу та адаптації контенту WEB-сайту для поліпшення користувацького досвіду.

Розроблена структура бази даних, яка використовує універсальне відношення для зберігання інформації про експерименти та їхні варіації. Ця структура бази даних дозволяє забезпечити ефективне зберігання та доступ до даних, не дублюючи інформацію.

Розроблено загальний алгоритм роботи програми, який визначає послідовність операцій, необхідних для створення експерименту та його варіацій. Цей алгоритм є основою для подальшого програмування та реалізації системи.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙЇНОЇ ТЕХНОЛОГІЇ A/B ТЕСТУВАННЯ ДЛЯ АДАПТАЦІЇ WEB-САЙТУ З ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

## 3.1 Варіантний аналіз і обґрунтування вибору мови та середовища програмування

З метою забезпечення виконання бажаних операцій апаратними компонентами комп'ютера, необхідно передати процесору чітко сформульовані програмні інструкції. Виконання цих інструкцій здійснюється за допомогою програмувальних мов, які нарешті перетворюються в машинний код, зрозумілий для процесора. Кожна програма складається з набору команд, які визначають операції, що повинні бути виконані апаратним обладнанням.

Машинний код це послідовність двійкових чисел (бітів) і може бути складним для розуміння і написання [16]. Однак для спрощення цього процесу використовуються високорівневі мови програмування. Вони дозволяють розробникам створювати і виконувати програми, нехтуючи особливостями архітектури процесора та не знаючи остаточного машинного коду. Кожна інструкція на високорівневій мові програмування складається з численних низькорівневих інструкцій, які в конечному підсумку будуть виконані. Вихідний код на таких високорівневих мовах зазвичай зрозумілий для розробника і годиться для подальшого редагування.

Перш ніж реалізувати програмне забезпечення для інформаційної технології проведення A/B тестування, необхідно детально проаналізувати наявні програмувальні мови і вибрати ту, яка найкраще відповідає потребам розробки програмного продукту. Сьогодні, написання таких додатків можливо на різних високорівневих мовах програмування. Тому важливо ретельно обрати програмувальну мову, таким чином, щоб розробка мультикомпонентної технології була найбільш оптимальною.

Для ретельного порівняння, було вибрано три з найбільш популярних мов програмування, які використовуються в WEB-розробці та загалом у програмуванні: TypeScript, C# та Kotlin.

TypeScript - це мова програмування, яка базується на JavaScript і надає можливість використовувати статичну типізацію для полегшення розробки WEB-додатків. Вона комбінує всі переваги JavaScript і додає сильну систему типізації, що допомагає виявляти помилки на ранніх етапах розробки. TypeScript має широку підтримку та численні сторонні бібліотеки, що робить її потужним інструментом для WEB-розробки та розробки загалом. На рисунку 3.1 подано ілюстрацію коду, написаного на мові TypeScript.

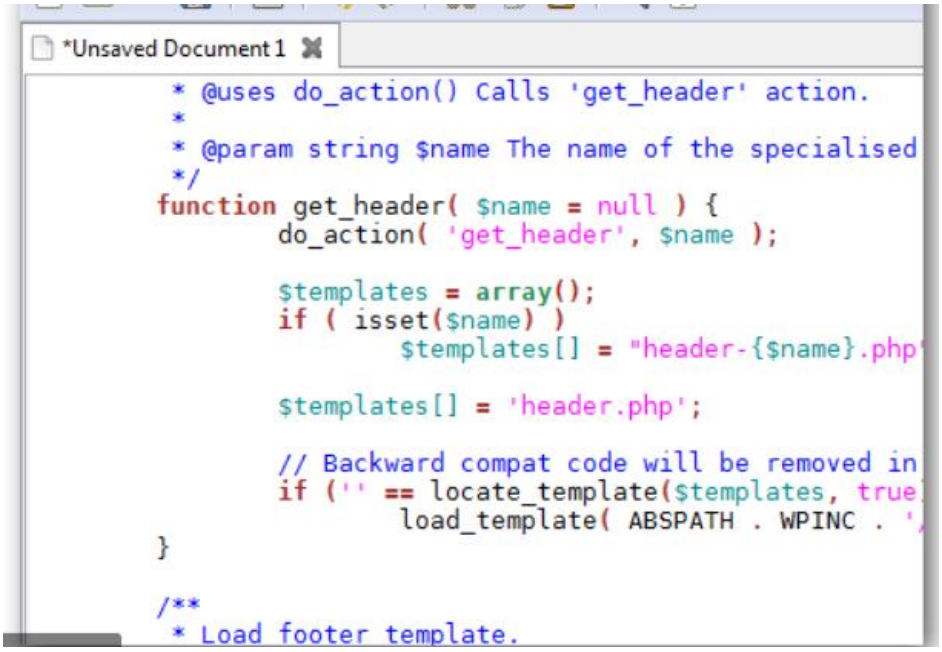



Рисунок 3.1 – Приклад коду на мові Typescript

C# - сучасна мова програмування, розроблена компанією Microsoft, яка використовується для розробки різноманітних програм, включаючи WEB-додатки. C# володіє рядом переваг [17], таких як підтримка компонент та статична типізація, що допомагають розробникам швидко та безпомилково створювати складні програми. Вона є мовою високого рівня зі схожим синтаксисом на C++ та Java. На рисунку 3.2 подано ілюстрацію коду, написаного на мові C#.

```coffeescript
fs = require 'fs'
express = require 'express'
app = express()

app.use express.bodyParser()
app.use app.router
app.use express.static(__dirname + '/public')
app.set 'views', __dirname + '/views'
app.set 'view engine', 'jade'

nouns = fs.readFileSync('nouns.txt', 'utf8').split('\n')

nurble = (text) ->
  text = text.toUpperCase()
  words = text.toLowerCase().replace(/[^a-z ]/g, '').split(' ')

  for word in words
    if word not in nouns
      re = new RegExp "(\\b)#{word}(\\b)","i"
      replacement = '$1<span class="nurble">nurble</span>$2'
      text = text.replace re, replacement

  text.replace  /\n/g, '<br>'

app.get '/', (req, res) ->
  res.render 'index'

app.post '/nurble', (req, res) ->
  res.render 'nurble', {nurble: nurble req.body.text}

app.listen 3005
```

Рисунок 3.2 – Приклад коду на мові C#

Kotlin - це мова програмування, яка отримала популярність в розробці Android-додатків та загалом в сфері програмування. Kotlin поєднує в собі переваги інших мов, таких як Java та C++, і надає розробникам зручний та безпечний інструмент для роботи. Вона має чіткий та зрозумілий синтаксис, а також підтримує функціональне програмування. На рисунку 3.3 подано ілюстрацію коду, написаного на мові Kotlin.

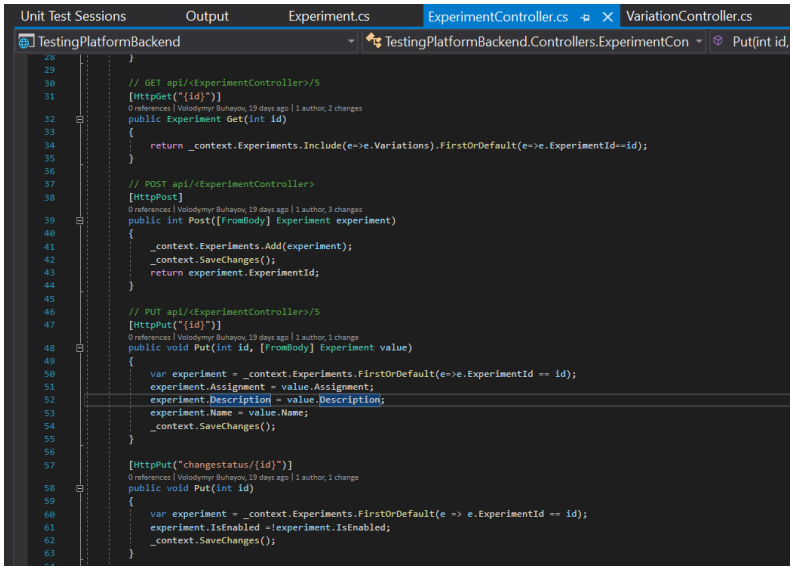



Рисунок 3.3 – Приклад коду на мові Kotlin

Кожна з цих мов має свої переваги та використовується у відповідних галузях розробки. З метою вибору мови для розробки платформи для A/B тестування, рекомендується створити таблицю порівняння, як показано у таблиці 3.1.

Таблиця 3.1 – Порівняння мов програмування

| Мова програмування | Types cript | C# | Kotlin |
|---|---|---|---|
| Придатність для клієнтської розробки | + | + | + |
| Придатність для серверної розробки | + | + | + |
| Статична типізація | + | + | + |
| Підтримка асинхронності | - | + | + |
| Інтегрована розробка | - | + | - |
| Загалом | 3 | 5 | 4 |

Після аналізу результатів порівняння можна визначити, що C# надає додатковий функціонал, який не мають інші порівнювані мови програмування. Таким чином, для створення серверної частини платформи рекомендовано вибрати мову C#.

Окрім вибору мови програмування, іншою ключовою складовою у процесі розробки програмного забезпечення є середовище розробки. Інтегроване середовище розробки (Integrated Development Environment або IDE) - це спеціалізований програмний інструмент, який надає розробникам комплексні засоби для створення, налагодження та підтримки програмного коду. Базові функціональні компоненти IDE включають текстовий редактор коду, автоматизовані засоби збирання та виконання коду, а також засоби відлагодження для виявлення та виправлення помилок.

Однак багато інтегрованих середовищ розробки йдуть далі та пропонують розширені можливості. Наприклад, вони можуть мати функціональні редактори коду, які виділяють синтаксис та надають рекомендації, вбудовані системи контролю версій для зберігання та відстежування змін в коді, інструменти для автоматизації процесів розробки, а також можливості інтеграції з іншими інструментами та розширеннями, що полегшують і прискорюють розробку програмного забезпечення.

Серед інтегрованих середовищ розробки, які дозволяють розробникам програмувати мовою C#, особливу популярність і визнання отримали Microsoft Visual Studio, Jetbrains Rider та Visual Studio Code. Ці інструменти не лише надають потужні засоби для розробки, але також підтримують багато корисних плагінів і розширень, що полегшують роботу розробників та підвищують продуктивність у процесі програмування.

Microsoft Visual Studio – це комплексний програмний продукт, який включає в себе інтегроване середовище розробки, а також розширений набір інструментів для розробки програмного забезпечення. Visual Studio розроблене компанією Microsoft і вже давно завоювало визнання серед програмістів та розробників завдяки своїй потужності та багатофункціональності. Це середовище надає широкий набір можливостей для створення різноманітних додатків, включаючи WEB-сайти, мобільні додатки, настільні програми та багато іншого.

Visual Studio володіє розширеними функціональними можливостями, включаючи інтелектуальне підсвічування синтаксису, підказки під час розробки, інтегровані системи керування версіями коду [18], а також інструменти для розладки та профілювання програм. Крім того, Visual Studio підтримує мови програмування, такі як C#, C++, Python, та інші, що робить його універсальним інструментом для різних проектів та технологій (рис. 3.4).

Однією з важливих переваг Visual Studio є його інтеграція з екосистемою Microsoft, що дозволяє розробникам легко взаємодіяти з

іншими продуктами та послугами, такими як Azure, .NET Framework, та багатьма іншими. Ця інтеграція спрощує розробку та публікацію програм для різних платформ та середовищ.

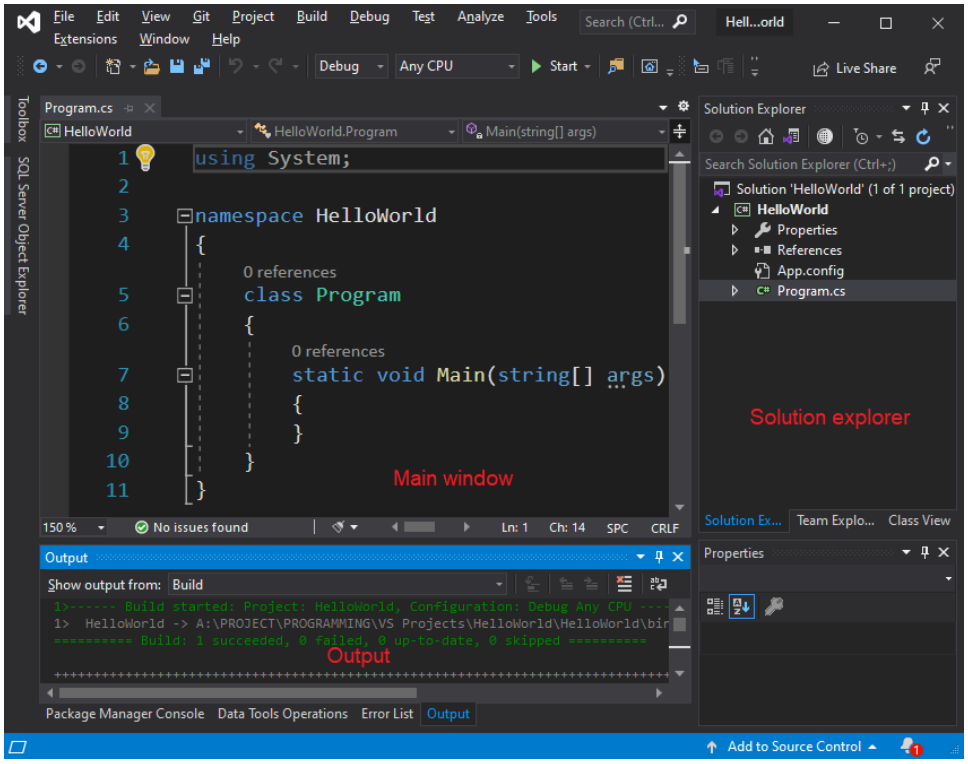



Рисунок 3.4 – Інтерфейс VS Studio

Варто відзначити, що Visual Studio надає можливість розробки як на платформі Windows, так і на інших операційних системах, що робить його доступним для широкого кола розробників. Таким чином, це потужне середовище розробки є однією з передових інструментальних платформ для створення високоякісного програмного забезпечення.

JetBrains Rider – це інше потужне інтегроване середовище розробки, яке вражає своєю ефективністю та розширеними можливостями для програмістів. Rider розроблене компанією JetBrains і стало важливим інструментом для розробників, які працюють з мовами програмування, такими як C#, F#, VB.NET, і багатьма іншими.

Це середовище розробки вражає інтегрованими засобами розладки, які допомагають виявляти та виправляти помилки в коді. Інтелектуальне

підсвічування синтаксису та автоматичні підказки роблять процес розробки більш продуктивним. JetBrains Rider також підтримує інтегровані системи керування версіями, що дозволяє зберігати та відстежувати зміни в коді.

Однією з ключових переваг Rider є його здатність працювати на різних платформах, включаючи Windows, macOS та Linux. Ця універсальність дозволяє розробникам вибирати оптимальну операційну систему для своїх потреб.

Додатково, Rider підтримує розширення та плагіни, що дозволяють розширити його функціональність та адаптувати під конкретні вимоги проекту (рис. 3.5). JetBrains Rider надає зручні інструменти для розробки різноманітних додатків, включаючи WEB-додатки, мобільні застосунки та інші види програмного забезпечення.

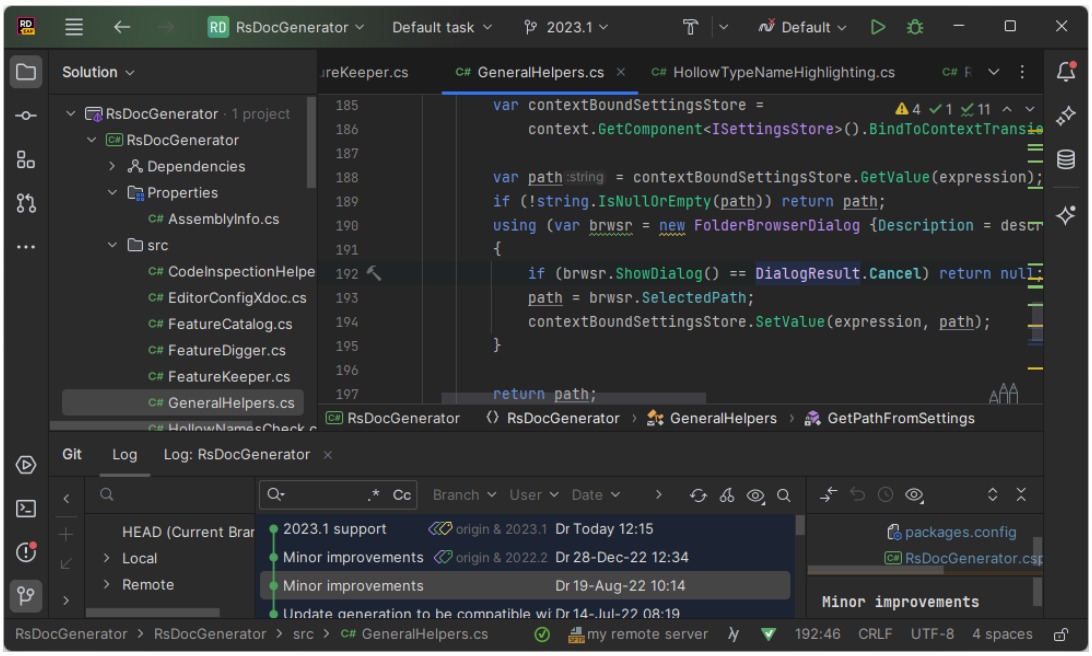



Рисунок 3.5 – Інтерфейс Rider

Інтеграція Rider з іншими інструментами та послугами, такими як ReSharper, TeamCity та Kotlin, робить його незамінним інструментом для розробників, які цінують продуктивність та якість свого коду. JetBrains Rider став неоціненим активом у світі розробки програмного забезпечення та допомагає розробникам досягати високих результатів у своїх проектах.

Visual Studio Code – це зручне і легковаге інтегроване середовище розробки, розроблене корпорацією Microsoft. Це вільний та відкритий програмний продукт, який здобув велику популярність серед розробників завдяки своїй легкості використання та розширюваності.

Однією з основних переваг Visual Studio Code є його універсальність, оскільки він підтримує різні мови програмування і платформи. Ви можете використовувати його для розробки WEB-додатків, мобільних застосунків, настільних програм та багато інших проектів. Це дозволяє розробникам використовувати одну і ту ж інтегровану середу розробки для різноманітних завдань.

Visual Studio Code має розширений набір інструментів для редагування коду, включаючи автодоповнення, інтелектуальне підсвічування синтаксису та велику кількість доступних розширень. Це допомагає розробникам писати ефективний та чистий код.

Інтегрована система керування версіями робить спільну роботу над проектами легкою та зручною. Visual Studio Code також має підтримку роботи з контейнерами, що полегшує розробку та впровадження додатків у різних середовищах.

Однією з важливих особливостей Visual Studio Code є його розширюваність. Розробники можуть створювати власні розширення та плагіни, які допомагають призначити роботу з IDE під конкретні потреби та вимоги проекту.

Visual Studio Code - це інтегрована середа розробки, яка пропонує багато можливостей та інструментів для розробників, і відзначається своєю легкістю використання та широким спектром підтримуваних мов та платформ.

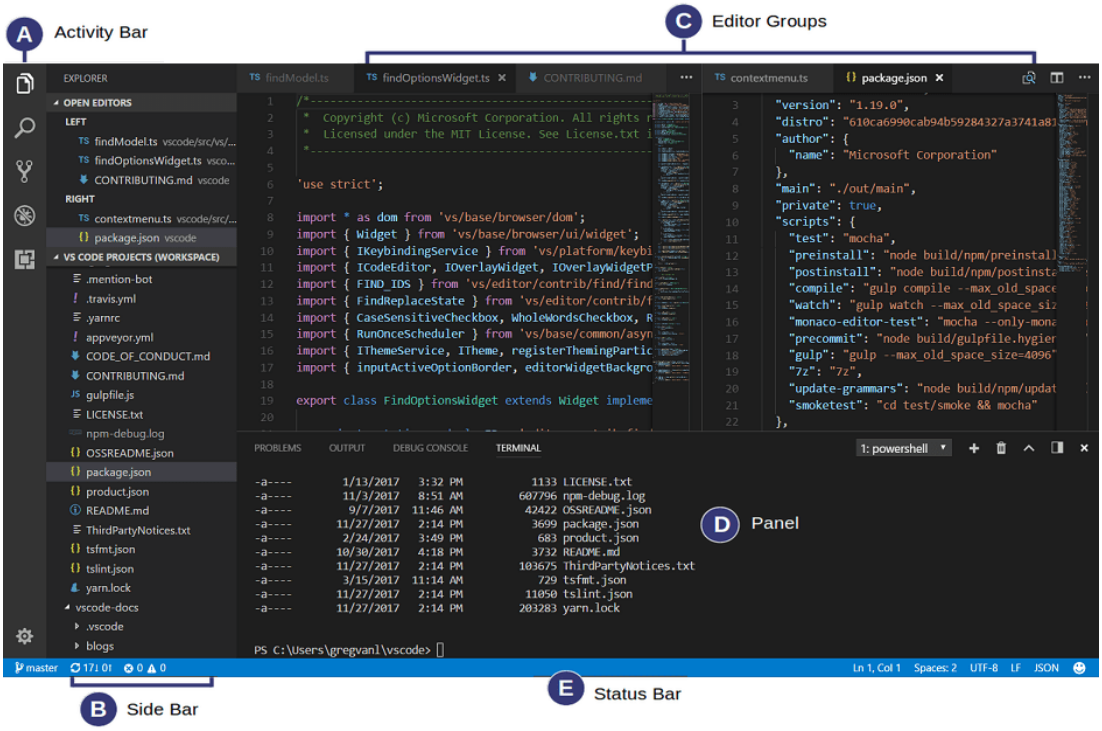



Рисунок 3.6 – Інтерфейс VS Code

В таблиці 3.2 подано порівняльний аналіз середовищ розробки.

Таблиця 3.2 – Порівняння мов програмування

| Мова програмування | VS Code | Visual Studio | Rider |
|---|---|---|---|
| Підтримка мов програмування | + | + | + |
| Розширюваність інструментів | + | + | + |
| Спрощена налагоджуваність | + | + | + |
| Інтегрована система керування версіями | - | + | + |
| Підтримка контейнерів | + | + | - |
| Загалом | 4 | 5 | 4 |

Після ретельного аналізу доступних інтегрованих середовищ розробки, було визначено, що оптимальним вибором для розробки технології є Microsoft Visual Studio IDE. Ця платформа забезпечує всі необхідні

інструменти для успішної реалізації інформаційної технології автоматизованого тестування. Крім того, вона надає можливості для подальшого розвитку та вдосконалення проекту завдяки своїй розширюваності та активній підтримці спільнотою розробників.

Отже, після аналізу та порівняння можливих варіантів, прийнято рішення використовувати високорівневу мову програмування C# та інтегроване середовище програмування Microsoft Visual Studio для реалізації інформаційної технології A/B тестування. Вони ідеально відповідають вимогам і потребам, які були визначені під час моделювання проекту. Крім того, ця комбінація має потенціал для подальшого розвитку та вдосконалення технології завдяки своїй гнучкості та широкому спектру можливостей.

## 3.2 Розробка модулів серверної частини інформаційної технології

В процесі моделювання, було визначено, що серверна частина складається з декількох основних модулей, а саме модуль керування експериментами, модуль роботи з великими мовними моделями та модулем призначення експерименту користувачу.

Модуль керування експериментами в програмі для проведення A/B тестування відіграє важливу роль у процесі розробки та виконання експериментів. Цей модуль включає в себе кілька ключових функцій, які спрощують весь процес.

Початково, модуль створення нового експерименту відіграє центральну роль при ініціалізації нового проекту. Він дозволяє користувачам вводити необхідні дані щодо експерименту, включаючи назву, опис і умови виконання. Ця інформація зберігається в базі даних для подальшого використання. Після успішного створення експерименту, користувач отримує унікальний ідентифікатор експерименту, який може бути використаний для посилання на нього.

Модуль керування експериментом надає користувачам можливість внесення змін в дані, які були введені при створенні експерименту. Крім того, він дозволяє змінювати розподіл трафіку, запускати та припиняти експеримент за потреби (рис. 3.7). Це допомагає користувачам ефективно керувати всіма аспектами свого експерименту.

```
  async function deleteVariation(event, variationId1){
   try {
    const response = await axios.delete(BACK_END_HOST_LOCATION + `/api/variation/${variationId1}`, axiosConfig);
    history.go(0)
   } catch (error) {
    alert(error);
   }
  }

  const [name, setName] = React.useState('');

  const handleNameChange = (event) => {
    setName(event.target.value);
  };
  const handleVariationCodeChange = (event) => {
   setVariationCode(event.target.value);
};
const handleVariationNameChange = (event) => {
 setVariationName(event.target.value);
};

  const [description, setDescription] = React.useState('');

  const handleDescriptionChange = (event) => {
    setDescription(event.target.value);
  };

  const [assignment, setAssignment] = React.useState('');

  const handleAssignmentChange = (event) => {
    setAssignment(event.target.value);
  };
```

Рисунок 3.7 –Модуль керування експериментом

Модуль керування експериментами також взаємодіє з модулем управління варіаціями, оскільки кожна варіація відноситься до певного експерименту. Користувачі можуть додавати нові варіації, змінювати їх назви та код, який виконується на сторінці при потраплянні користувача на конкретну варіацію. Це робить процес керування варіаціями дуже зручним та гнучким.

Важливо зазначити, що доступ до модуля керування експериментами надається лише зареєстрованим користувачам, що забезпечує безпеку та

контроль за роботою з експериментами. Крім того, інформація про користувачів, які вносять зміни за допомогою цього модуля, зберігається для подальшого перегляду та аналізу. Це допомагає зберігати історію роботи з експериментами та робити відповідальність за керування проектом більш прозорою і дієвою.

Модуль роботи з великими мовними моделями в програмі є важливим компонентом, що дозволяє використовувати потужність мовних моделей для обробки текстових даних. Основною ідеєю цього модуля є можливість використовувати плейсхолдери, такі як {{test question?}} наприклад, в JavaScript-коді для динамічного взаємодії з великими мовними моделями (рис. 3.8).

```
src > pages > ListOfExperiments.tsx > ListOfExperiments > getExperiments > response.data.map() callback
      You, 2 days ago | 1 author (You)
1     import { Box, Button, Divider, List, ListItem, ListItemButton, ListItemIcon, ListItemText, TextField } from '@mui/material'
2     import React, { Fragment, useEffect, useState } from 'react'
3     import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
4     import './CreateExperimentStyle.css';
5     const axios = require('axios').default;
6
7     export const ListOfExperiments: React.FC = () => {
8       const [listOfExperiments, setListOfExperiments] = useState([]);
9
10      async function getExperiments() {
11        try {
12          const response = await axios.get(BACK_END_HOST_LOCATION + '/api/experiment', axiosConfig);
13          setListOfExperiments(response?.data?.map(experiment =>  {
14              return <Fragment>
15             <ListItem disablePadding>
16               <ListItemButton  component="a" href={`/experiment/${experiment.experimentId}`}>
17              <ListItemText primary={experiment.experimentId} />      You, 2 days ago • fix
18               <ListItemText primary={experiment.name} />
19             </ListItemButton>
20           </ListItem>
21                       <Divider />
22         </Fragment>
23         }
24       ))
25         } catch (error) {
26           alert(error);
27         }
28       }
29       useEffect(() => {
30         getExperiments();
31        },[]);
32      return (
33       <Fragment>
34           <Box sx={{ width: '70%', bgcolor: 'background.paper' }}>
35         <nav aria-label="main mailbox folders">
36           <List>
37           <Divider />
38           {listOfExperiments}
39           </List>
40           </nav>
41           </Box>
42       </Fragment>
43      )
44    }
45
```

Рисунок 3.8 –Модуль роботи з великими мовними моделями

Під час використання цього модуля, користувач може вставляти плейсхолдери в свій JavaScript-код. Коли виконується код, програма автоматично вилучає текст з цих плейсхолдерів та передає його до визначеного вендора великих мовних моделей. Вендор використовує цей текст для генерації відповіді, яка потім повертається в програму.

Отримавши відповідь від вендора, програма автоматично замінює плейсхолдери у вихідному тексті на дані, які отримала від вендора. Це дозволяє користувачеві отримувати готовий текст з обробкою, проведеною великою мовною моделлю, без необхідності розуміння її роботи в деталях.

Крім того, модуль роботи з великими мовними моделями може виконувати додатковий код, який користувач вклав у свої плейсхолдери, застосовуючи дані, які були оброблені великою мовною моделлю. Це робить модуль дуже потужним і зручним інструментом для роботи з текстом та великими мовними моделями.

Модуль призначення експерименту користувачу є важливою складовою системи для проведення A/B тестування. Його головна мета полягає в тому, щоб визначити, чи варто призначити конкретному користувачу участь у певному експерименті на основі заданих умов (рис. 3.9).

Цей модуль дозволяє системі виконувати код в браузері користувача та аналізувати, чи задовольняють вони певні умови, які прив'язані до конкретного експерименту. Якщо умови виконуються, користувачу призначається цей експеримент. Це важливо для забезпечення об'єктивності та точності результатів A/B тестування.

```
export const CreateExperiment: React.FC = () => {
  let history = useHistory();

  async function createExperiment() {
    try {
      const data = { name: name, description: description, assignment: assignment};
      const response = await axios.post(BACK_END_HOST_LOCATION + '/api/experiment', data, axiosConfig);
      history.push(`/experiment/${response.data}`);
      history.go(0)
    } catch (error) {
      alert(error);
    }
  }

  const [name, setName] = React.useState('');

  const handleNameChange = (event) => {
    setName(event.target.value);
  };

  const [description, setDescription] = React.useState('');

  const handleDescriptionChange = (event) => {
    setDescription(event.target.value);
  };

  const [assignment, setAssignment] = React.useState('');

  const handleAssignmentChange = (event) => {
    setAssignment(event.target.value);
  };

  const onCreateButtonClick = () => {
    createExperiment();
  }

 return (
  <Fragment>
    <div className="createExperiment_container">
    <TextField id="outlined-basic" label="Назва" variant="outlined"  sx={{marginBottom: "20px", width: "65%"}}
               value={name}
               onChange={handleNameChange} />
    <TextField
        id="outlined-multiline-static"
        label="Опис"
        multiline
        rows={4}
        sx={{marginBottom: "20px", width: "65%"}}
        value={description}
        onChange={handleDescriptionChange}
      />
          <TextField
        id="outlined-multiline-static"
        label="Умова попадання на експеримент (JavaScript)"
        multiline
```

Рисунок 3.9–Модуль призначення експерименту

Завдяки модулю призначення експерименту користувачу, можна забезпечити, що кожен користувач бере участь у тестуванні, яке найкраще відповідає його профілю та контексту. Такий підхід допомагає отримати більш точні та корисні результати з A/B тестів.

Для користувача цей модуль прозорий – він не має прямої взаємодії з ним і не потребує від користувача жодних дій. Однак він грає ключову роль в забезпеченні того, що експерименти призначаються об'єктивно та згідно з встановленими критеріями.

Підсумовуючи результати розробки серверної частини інформаційної технології для проведення A/B тестування, можна відзначити успішну реалізацію трьох важливих модулів, які включають загальну логіку керування експериментами, роботу з великими мовними моделями та призначення експериментів. Програмні функції, що були визначені на етапі моделювання були успішно впроваджені під час розробки компонент.

### 3.3 Розробка модулів клієнтської частини інформаційної технології

Клієнтська частина використовує платформу React для побудови модулів та відображення їх в браузері клієнта.

Реакт (React) - це відкрите програмне забезпечення JavaScript-фреймворк для розробки користувацького інтерфейсу, який був створений та підтримується компанією Facebook. Він широко використовується для створення WEB-додатків та інтерфейсів з високою реактивністю і відгуком користувача.

Однією з ключових переваг React є використання віртуального DOM (Document Object Model). Це дозволяє оптимізувати взаємодію з реальним DOM та зменшити кількість зайвих операцій, що робляться для оновлення сторінки. Це полегшує створення швидких та реактивних WEB-додатків.

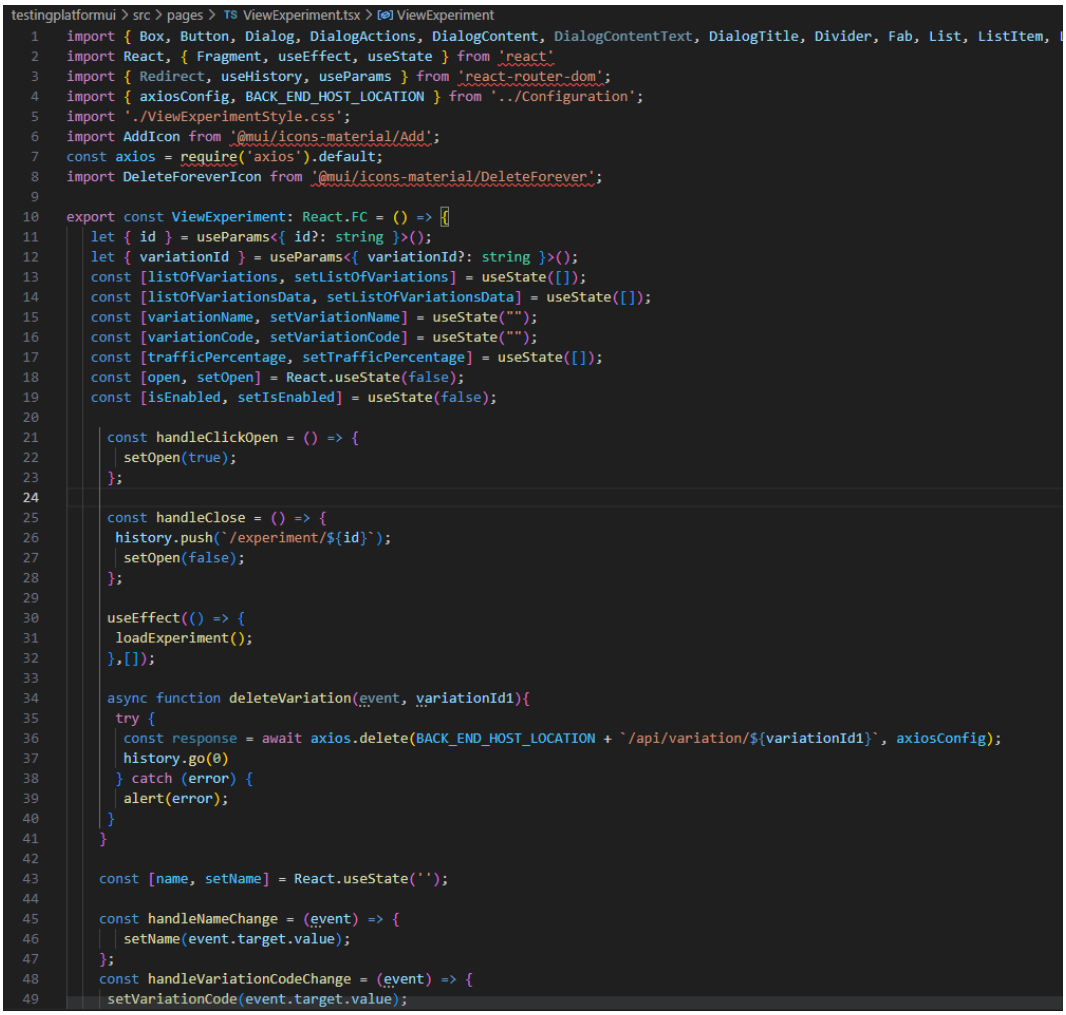
React працює на основі компонентів, які можна створювати, комбінувати та використовувати повторно. Це спрощує розподілення функціональності та покращує рефакторинг коду. Ви можете розглядати компоненти, як окремі будівельні блоки для вашого додатку.

React можна поєднувати з іншими бібліотеками та фреймворками, такими як Redux або React Router, щоб розширити його можливості. Ви можете легко інтегрувати React у вже існуючі проекти.

Клієнтська частина містить декілька основних модулів, а саме модуль керування експериментом, модуль керування всіма експериментами, модуль

розподілення користувачів на експерименті та модуль створення нового експерименту.

Модуль відображення інформації про експеримент у клієнтській частині є важливою складовою для користувачів, які відповідають за проведення A/B тестування (рис 3.10).



```
testingplatformui > src > pages > TS ViewExperiment.tsx > [@] ViewExperiment
import { Box, Button, Dialog, DialogActions, DialogContent, DialogContentText, DialogTitle, Divider, Fab, List, ListItem,
import React, { Fragment, useEffect, useState } from 'react'
import { Redirect, useHistory, useParams } from 'react-router-dom';
import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
import './ViewExperimentStyle.css';
import AddIcon from '@mui/icons-material/Add';
const axios = require('axios').default;
import DeleteForeverIcon from '@mui/icons-material/DeleteForever';

export const ViewExperiment: React.FC = () => {
  let { id } = useParams<{ id?: string }>();
  let { variationId } = useParams<{ variationId?: string }>();
  const [listOfVariations, setListOfVariations] = useState([]);
  const [listOfVariationsData, setListOfVariationsData] = useState([]);
  const [variationName, setVariationName] = useState("");
  const [variationCode, setVariationCode] = useState("");
  const [trafficPercentage, setTrafficPercentage] = useState([]);
  const [open, setOpen] = React.useState(false);
  const [isEnabled, setIsEnabled] = useState(false);

    const handleClickOpen = () => {
      setOpen(true);
    };

    const handleClose = () => {
     history.push(`/experiment/${id}`);
      setOpen(false);
    };

    useEffect(() => {
     loadExperiment();
    },[]);

    async function deleteVariation(event, variationId1){
     try {
      const response = await axios.delete(BACK_END_HOST_LOCATION + `/api/variation/${variationId1}`, axiosConfig);
      history.go(0)
     } catch (error) {
      alert(error);
    }
   }

  const [name, setName] = React.useState('');

  const handleNameChange = (event) => {
      setName(event.target.value);
  };
  const handleVariationCodeChange = (event) => {
   setVariationCode(event.target.value);
```

Рисунок 3.10 – Модуль керування експериментом

Цей модуль надає доступ до усіх необхідних функцій для керування експериментом та його варіаціями, що включають наступні можливості:

1. Створення нової варіації: користувач має можлисть створити нову варіацію для вибраного експерименту. Це включає в себе задання назви варіації, опису та інших параметрів.

2. Зміна розподілу трафіку: модуль дозволяє користувачам налаштовувати співвідношення трафіку між різними варіаціями експерименту. Вони можуть змінювати це співвідношення залежно від потреб і вимог експерименту.

3. Зміна умов таргетингу: користувачі можуть налаштовувати умови, за якими варіації експерименту будуть показуватися користувачам. Це дозволяє проводити тестування на специфічних групах аудиторії для отримання більш точних результатів.

4. Запуск та зупинення експерименту: модуль надає можливість запускати та зупиняти експерименти з відображенням відповідних статусів. Це робить процес керування експериментами більш зручним та контрольованим.

5. Видалення варіації та експерименту: користувачі мають можливість видаляти непотрібні варіації та експерименти з системи. Це дає можливість підтримувати порядок у списку експериментів та зменшувати зайве навантаження на систему.

Загалом, цей модуль відображення інформації про експеримент дозволяє користувачам ефективно керувати та моніторити всіма аспектами A/B тестування, забезпечуючи зручність та точність проведення тестів.

Модуль керування всіма експериментами є центральним інструментом для адміністрації та моніторингу всіх активних та завершених експериментів (рис. 3.11).

```
4
5   export const Navbar: React.FC = () => (
6   <Fragment>
7   <Box sx={{ flexGrow: 1, marginBottom: "33px" }}>
8         <AppBar position="static">
9           <Toolbar>
10            <Typography variant="h6" component="div" sx={{ flexGrow: 1 }}>
11            Платформа управління A/B тестами
12            </Typography>
13            <Button color="inherit" href="/list">Список експериментів</Button>
14            <Button color="inherit" href="/createexperiment">Створити новий</Button>
15          </Toolbar>
16        </AppBar>
17      </Box>
18  </Fragment>
19  )
```

Рисунок 3.11 – Модуль керування всіма експериментами

Цей модуль забезпечує важливі функції для керування експериментами та їх параметрами, включаючи наступні можливості:

1. Відображення списку всіх експериментів: модуль надає користувачам загальний перелік усіх створених експериментів, показуючи їх назви, стани (активний чи завершений) та інші параметри.

2. Активність та статус експериментів: користувачі можуть швидко перевіряти активність кожного експерименту, а також його поточний стан (наприклад, "активний," "завершений," тощо).

3. Можливість пошуку та фільтрації: модуль дозволяє користувачам шукати конкретні експерименти за їх назвою та застосовувати фільтри для групування за станом (наприклад, показати лише активні експерименти).

4. Створення нового експерименту: цей модуль також надає можливість створити новий експеримент, додавши інформацію про назву, опис та інші параметри. Важливою є зручність та простота процесу створення нових експериментів.

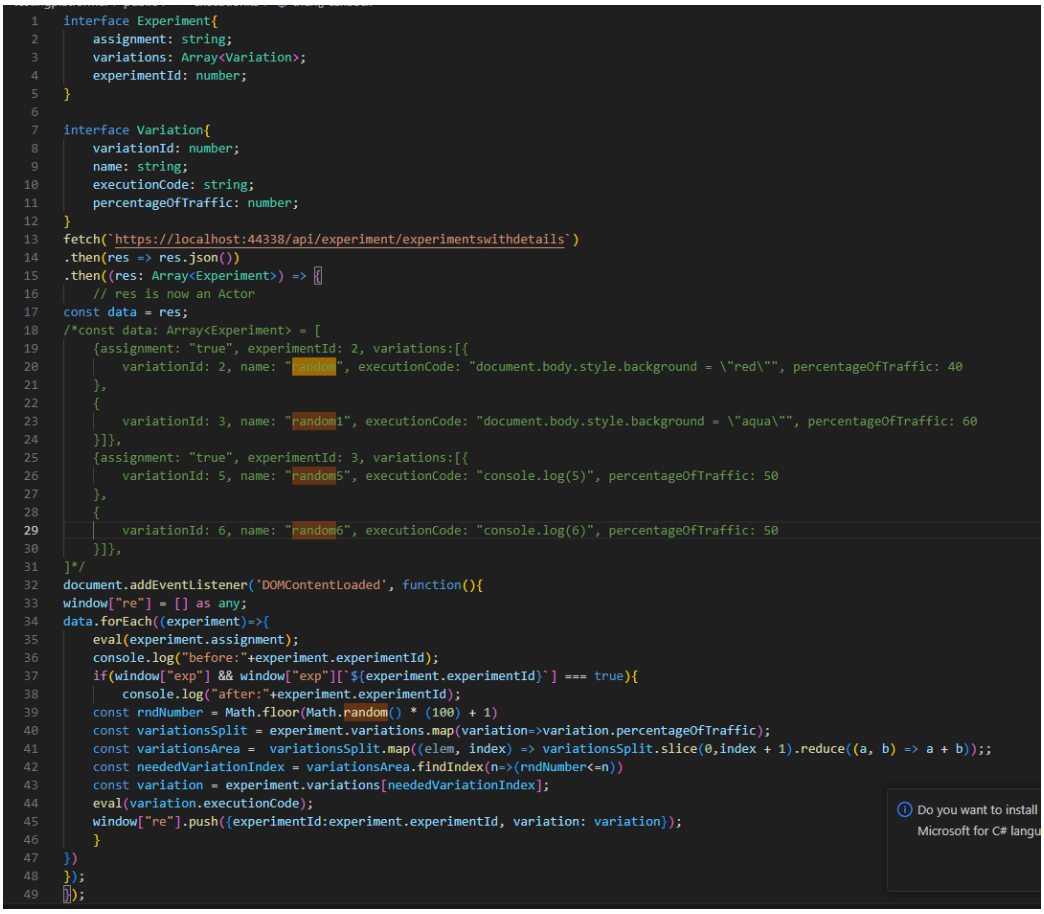
5. Управління параметрами експерименту: користувачі можуть змінювати параметри і налаштування для кожного окремого експерименту, такі як умови таргетингу, розподіл трафіку, активність та інші параметри.

6. Відображення статистики та результатів: модуль надає користувачам можливість переглядати статистику та результати кожного експерименту, що допомагає в прийнятті інформованих рішень та аналізі результатів тестування.

Загалом, модуль керування всіма експериментами створений для спрощення та зручного управління великою кількістю експериментів в системі, забезпечуючи адміністраторам та користувачам швидкий та зручний доступ до всієї необхідної інформації.

Модуль розподілення користувачів на експерименті є важливою складовою системи тестування A/B. Його основна функція полягає в тому, щоб ефективно розподіляти користувачів між різними варіаціями

експерименту згідно з параметрами, встановленими для кожної варіації (рис. 3.12).



```
interface Experiment{
    assignment: string;
    variations: Array<Variation>;
    experimentId: number;
}

interface Variation{
    variationId: number;
    name: string;
    executionCode: string;
    percentageOfTraffic: number;
}
fetch(`https://localhost:44338/api/experiment/experimentswithdetails`)
.then(res => res.json())
.then((res: Array<Experiment>) => {
    // res is now an Actor
const data = res;
/*const data: Array<Experiment> = [
    {assignment: "true", experimentId: 2, variations:[{
        variationId: 2, name: "random", executionCode: "document.body.style.background = \"red\"", percentageOfTraffic: 40
    },
    {
        variationId: 3, name: "random1", executionCode: "document.body.style.background = \"aqua\"", percentageOfTraffic: 60
    }]},
    {assignment: "true", experimentId: 3, variations:[{
        variationId: 5, name: "random5", executionCode: "console.log(5)", percentageOfTraffic: 50
    },
    {
        variationId: 6, name: "random6", executionCode: "console.log(6)", percentageOfTraffic: 50
    }]},
]*/
document.addEventListener('DOMContentLoaded', function(){
window["re"] = [] as any;
data.forEach((experiment)=>{
    eval(experiment.assignment);
    console.log("before:"+experiment.experimentId);
    if(window["exp"] && window["exp"][`${experiment.experimentId}`] === true){
        console.log("after:"+experiment.experimentId);
    const rndNumber = Math.floor(Math.random() * (100) + 1)
    const variationsSplit = experiment.variations.map(variation=>variation.percentageOfTraffic);
    const variationsArea =  variationsSplit.map((elem, index) => variationsSplit.slice(0,index + 1).reduce((a, b) => a + b));;
    const neededVariationIndex = variationsArea.findIndex(n=>(rndNumber<=n))
    const variation = experiment.variations[neededVariationIndex];
    eval(variation.executionCode);
    window["re"].push({experimentId:experiment.experimentId, variation: variation});
    }
})
});
});
```

Рисунок 3.12 – Модуль розподілення користувачів

Основні можливості модуля розподілення користувачів на експерименті включають:

1. Розподіл трафіку за параметрами експерименту: модуль отримує параметри експерименту, такі як розподіл трафіку між варіаціями, та дотримується цих параметрів при розподіленні користувачів.

2. Генерація випадкових чисел: модуль використовує генерацію випадкових чисел у масштабі від 0 до 100 для кожного користувача, щоб визначити, до якої варіації експерименту вони належать. Це забезпечує об'єктивний та випадковий розподіл користувачів.

3. Визначення приналежності до варіації: після генерації випадкового числа модуль визначає, до якої варіації користувач належить згідно з заданими розподільними параметрами.

4. Виконання відповідного коду: коли варіація експерименту визначена для конкретного користувача, відповідний код, пов'язаний з цією варіацією, виконується в браузері користувача. Це дозволяє проводити тестування та аналізувати вплив різних варіацій на поведінку користувачів.

5. Збереження результатів розподілу: модуль зберігає інформацію про те, як користувачі були розподілені між варіаціями експерименту. Це допомагає в подальшому аналізі та оцінці результатів тестування.

Загалом, модуль розподілення користувачів відіграє ключову роль у проведенні A/B тестів, забезпечуючи чесний та випадковий розподіл користувачів між різними варіаціями експерименту. Це допомагає зібрати об'єктивні дані та оцінити вплив змін на користувачів.

Модуль створення нового експерименту є ключовою складовою системи проведення A/B тестування. Він надає користувачам можливість ініціювати новий експеримент і визначити його основні параметри (рис. 3.13).

```
import { Button, TextField } from '@mui/material'
import React, { Fragment } from 'react'
import { useHistory } from 'react-router-dom';
import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
import './CreateExperimentStyle.css';
const axios = require('axios').default;

export const CreateExperiment: React.FC = () => {
    let history = useHistory();

    async function createExperiment() {
        try {
            const data = { name: name, description: description, assignment: assignment};
          const response = await axios.post(BACK_END_HOST_LOCATION + '/api/experiment', data, axiosConfig);
          history.push(`/experiment/${response.data}`);
          history.go(0)
        } catch (error) {
          alert(error);
        }
      }

    const [name, setName] = React.useState('');

    const handleNameChange = (event) => {
        setName(event.target.value);
    };

    const [description, setDescription] = React.useState('');

    const handleDescriptionChange = (event) => {
        setDescription(event.target.value);
    };

    const [assignment, setAssignment] = React.useState('');

    const handleAssignmentChange = (event) => {
        setAssignment(event.target.value);
    };

    const onCreateButtonClick = () => {
        createExperiment();
    }
```

Рисунок 3.13 – Модуль створення нового користувача

Розглянуто детальніше функції та можливості цього модуля:

1. Введення загальної інформації: першим кроком при створенні нового експерименту користувач вводить загальну інформацію. Ця інформація може включати назву експерименту, короткий опис, цільові метрики та інші ключові параметри, що описують експеримент.

2. Визначення варіацій: після введення загальної інформації користувач має можливість визначити різні варіації для експерименту. Варіації представляють різні версії або варіанти тестування. Для кожної варіації можна задати назву та внести специфічні параметри, які будуть тестуватися.

3. Розподіл трафіку: користувач може задати, як розподіляти трафік між різними варіаціями. Налаштування розподілу трафіку допомагає визначити, скільки користувачів буде спрямовано до кожної варіації.

4. Умови виконання: модуль також дозволяє визначити умови, при яких експеримент буде виконуватися. Це може бути включення чи виключення певних користувачів за допомогою таргетингу.

5. Попередні перевірки та збереження: після введення всіх параметрів користувач має можливість попередньо переглянути і перевірити конфігурацію експерименту перед створенням. Після чого він може підтвердити створення нового експерименту.

6. Збереження та унікальний ідентифікатор: після створення експерименту, в системі зберігається унікальний ідентифікатор експерименту, який може бути використаний для посилання на нього та управління ним у майбутньому.

7. Посилання на керування експериментом: після створення нового експерименту користувач отримує можливість перейти до модуля керування експериментом, де він може змінювати параметри, розподіл трафіку, стартувати та зупиняти експеримент, а також додавати, редагувати чи видаляти варіації.

Модуль створення нового експерименту є важливою складовою системи A/B тестування, оскільки від нього починається процес створення та

налаштування експериментів. Він допомагає користувачам визначити параметри експерименту та готувати його до подальшого тестування та аналізу.

У цьому розділі були розглянуті клієнтські компоненти системи проведення A/B тестування. Вони включають модуль відображення інформації про експерименти, який надає користувачам можливість управляти параметрами та станом експериментів. Крім того, модуль керування всіма експериментами дозволяє здійснювати пошук, групування та створення нових експериментів. Завершує цей розділ модуль розподілення користувачів на експериментах, який визначає, яким користувачам будуть надані різні варіації тесту. Ці компоненти важливі для створення та управління A/B експериментами та допомагають користувачам налаштовувати та контролювати їхні тести зручно та ефективно.

## 3.4 Тестування та аналіз результатів роботи програмного забезпечення

Тестування програмного продукту є ключовим етапом в процесі розробки, спрямованим на перевірку та підтвердження відповідності функціональності додатку вимогам та очікуванням. Цей процес передбачає виконання ряду програмних або системних компонент з метою виявлення помилок, невідповідності вимогам та потенційних проблем в системі.

Одним з методів тестування є автоматизоване тестування, яке передбачає використання спеціалізованих систем для автоматизації процесу перевірки програмного продукту. Це дозволяє виконувати тести ефективніше та прискорює процес виявлення та виправлення помилок.

Для забезпечення автоматизованого тестування проекту будуть використовуватися unit-тести, оскільки вони характеризуються стабільністю та ефективністю [19]. Цей вид тестів може бути розроблений у тому ж середовищі, в якому проводиться основна розробка програми, і не потребує

великої кількості часу для виконання. Крім того, завдяки додатковим розширенням, можливо налаштувати автоматичне виконання тестів при кожній модифікації коду.

Автоматичне виконання тестів під час розробки дозволяє розробникам оперативно виявляти та виправляти помилки, оскільки вони будуть сповіщені про них вразі виникнення. Ця функціональність особливо корисна при використанні методології розробки через тести (Test-Driven Development, TDD), яка передбачає написання тестів перед власне розробкою коду. Такий підхід допомагає забезпечити якість програмного продукту та покращити процес розробки.

На сьогоднішній день існує значна різноманітність бібліотек для виконання unit-тестів у мові програмування C#. Для вибору найбільш підходящого варіанту, варто розглянути та порівняти три найпопулярніші бібліотеки, які активно використовуються на ринку. Детальний аналіз фреймворків для unit-тестування представлено в таблиці 3.3.

Таблиця 3.3 – Порівняльна характеристика платформ для unit тестування

| Назва | NUnit | XUnit | MSTest |
|---|---|---|---|
| Параметризовані тести | + | + | - |
| Кастомні атрибути тестів | + | + | + |
| Можливість отримати доступ до тестового контексту | + | + | + |
| Запуск на різних платформах | + | + | + |
| Асинхронна підтримка | + | + | + |
| Підтримка мокінгу | + | - | + |
| Загалом | 6 | 5 | 5 |

Порівнюючи бібліотеки NUnit, MSTest та xUnit для unit-тестування, можна зазначити, що всі вони мають подібні функціональність та

підтримують популярні характеристики. Однак бібліотека NUnit відрізняється активною спільнотою підтримки та багатома розширеннями, що роблять її дуже потужним інструментом для unit-тестування. Особливо важливо відзначити підтримку мокінгу та можливість запуску тестів на різних платформах. Отже, в результаті цього порівняння, бібліотека NUnit може бути відмінним вибором для unit-тестування завдяки своїм розширеним можливостям та активній спільноті підтримки.

Розглянемо юніт тест, що відповідає за тестування методу, що здійснює пошук експерименту по його назві, що юзер може задати в пошуку. Такий тест має на вхід приймати стрічку з назвою експерименту. Після цього створювати об’єкт на основі классу ExperimentController та викликати один з його методів GetExperimentByName. Оскільки даний метод здійснює виклик до бази даних, варто застосувати інструменти мокінгу, а саме мокінг виклику до бази даних. Це дасть змогу перевірити, що дійсно виклик має здійснюватися, без фактичного виконання виклику бази даних, як результат цього це висока стабільність написаних тестів.

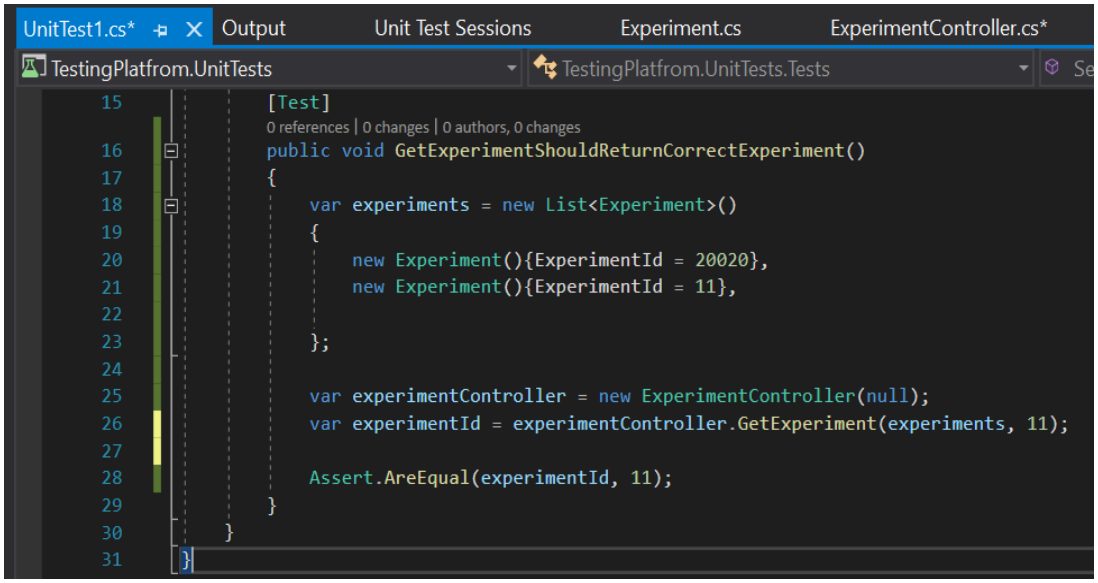



Рисунок 3.14 – Unit тест для пошуку експерименту

Варто також, здійснити комплексне тестування додатку за допомогою ручного тестування. Для цього буде використовуватися браузер Opera останьої версії.

Сценарій тестування №1 спрямований на перевірку можливості зміни розподілу трафіку в системі. Цей тест допомагає визначити, чи працює функціонал зміни розподілу трафіку на платформі для керування експериментами.

Опис кроків тестування:

1. Створити вкладку браузера.
2. Перейти на WEB-сайт, де проводиться керування експериментами.
3. На головній сторінці вибрати будь-який експеримент.
4. Внести зміни до даних з розподілом трафіку.
5. Натиснути кнопку «Зберегти».

Очікуваний результат полягає в перезавантажені сторінки користувача з відображенням нових відсотків розподілу.

На рисунку 3.15 можна побачити результат виконання тест-кейсу, який відповідає очікуваному результату, підтверджуючи коректну роботу функціоналу.

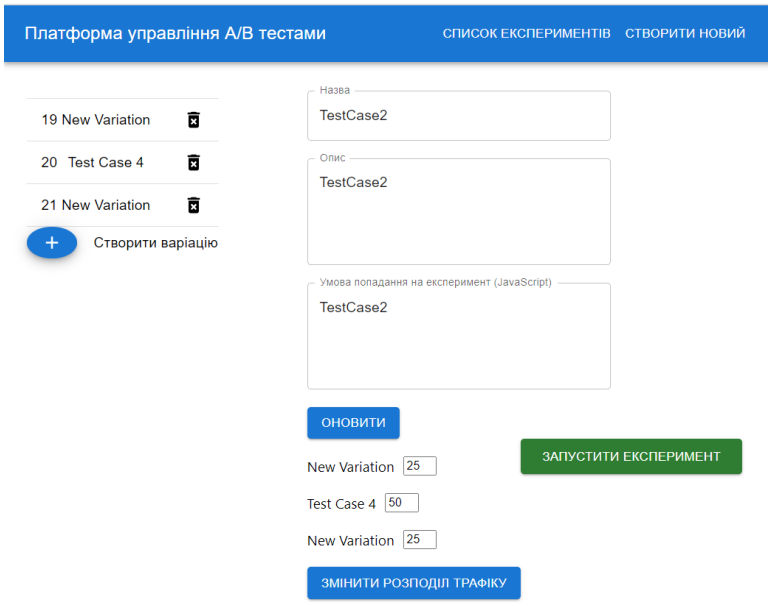



Рисунок 3.15 – Результат сценарію №1

Сценарій тестування №2 спрямований на перевірку можливості створоення нового експерименту в системі. Цей тест допомагає визначити, чи працює функціонал зміни створення експерименту на платформі для керування експериментами.

Опис кроків тестування:

6. Створити вкладку браузера.
7. Перейти на WEB-сайт, де проводиться керування експериментами
8. На головній сторінці натиснути кнопку "Створити новий експеримент".
9. Ввести необхідні дані, такі як назва експерименту, опис та умови потрапляння.
10. Натиснути кнопку «Зберегти».

Очікуваний результат полягає в переадресації користувача на сторінку зі створеним експериментом та внесеними даними. Ця переадресація вказує на успішне створення експерименту та коректну роботу функціоналу для зміни розподілу трафіку.

На рисунку 3.16 можна побачити результат виконання тест-кейсу, який відповідає очікуваному результату, підтверджуючи коректну роботу функціоналу.

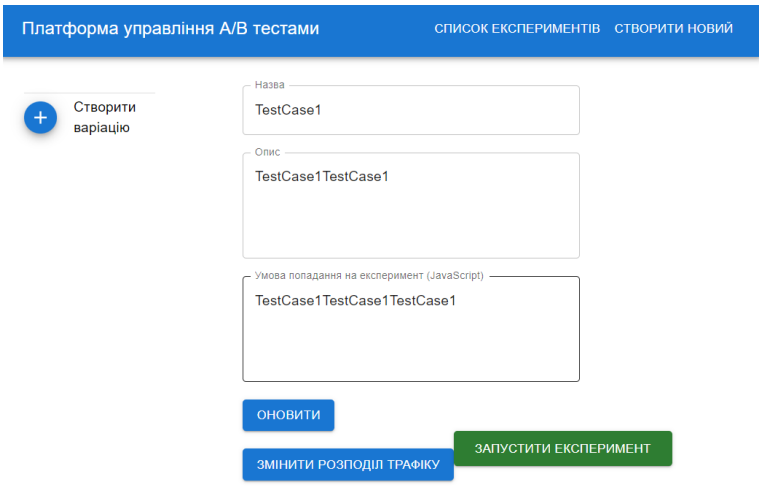



Рисунок 3.16 – Результат сценарію №2

Ще одним аспектом, який важливий для оцінки системи, є її продуктивність у виконанні процесу тестування.

Тестування швидкодії для платформи проведення A/B експериментів включає в себе оцінку продуктивності системи при виконанні різних операцій та завдань. Основною метою цього тестування є визначення, наскільки швидко та ефективно платформа може виконувати операції, необхідні для проведення A/B тестування.

Для тестування швидкодії використовуються різні типи завдань та операцій, які реалізовані в системі. Деякі з них можуть включати:

1. Створення нового експерименту: Перевірка, як швидко система може обробити запит на створення нового експерименту, включаючи введення всіх необхідних даних та збереження їх у базі даних.

2. Зміна розподілу трафіку: Визначення, наскільки оперативно система може оновити розподіл трафіку для існуючого експерименту, забезпечуючи коректну роботу системи.

3. Запуск або зупинення експерименту: Вимірювання часу, необхідного для активації або вимкнення певного експерименту.

4. Пошук та сортування експериментів: Оцінка продуктивності під час пошуку та фільтрації експериментів за різними параметрами, такими як назва, стан, або інші атрибути.

5. Виконання варіантів експерименту: Визначення часу, необхідного для обчислення та відправки відповідних варіантів експерименту користувачам.

6. Видалення експериментів та варіацій: Тестування швидкодії під час видалення експериментів та варіацій з системи.

7. Завантаження системи при великому навантаженні: Симуляція великої кількості одночасних запитів та перевірка, як система управляє навантаженням.

Важливим аспектом тестування швидкодії є визначення середнього часу виконання кожної операції та їх максимального часу виконання. Це допомагає ідентифікувати можливі проблеми з продуктивністю та шукати шляхи їх оптимізації. Також важливо враховувати, як система веде себе при великому обсязі даних та навантаженні.

Для перевірки швидкодії було створено нову збірку з запитами у програмі Postman, що перевіряє час виконання експерименту на сторінці. Результати швидкодії наведено на рисунку 3.17.

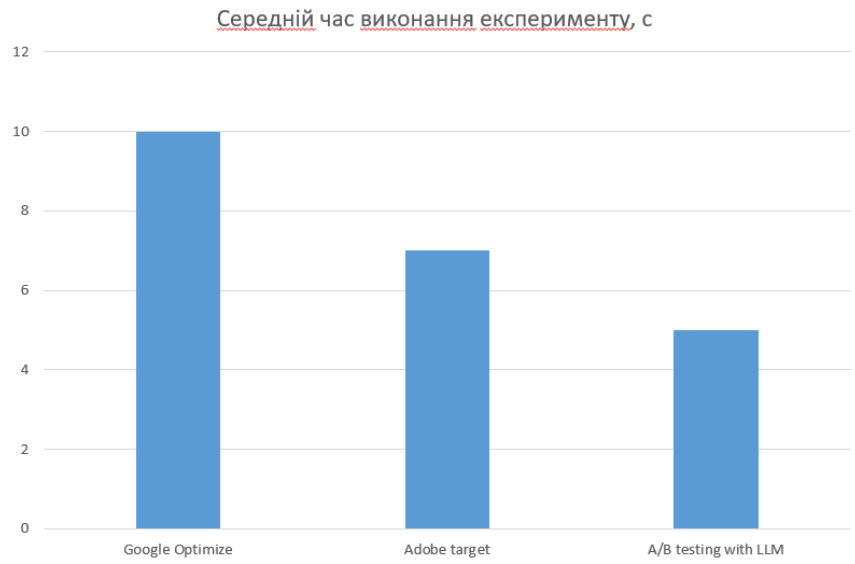



Рисунок 3.17 – Результати швидкодії

Тестування швидкодії для A/B платформи допомагає переконатися, що система здатна ефективно обробляти всі необхідні операції для проведення тестування та забезпечити користувачам максимально комфортний досвід взаємодії з нею. Результати швидкодії відображають, що розроблювальна платформа швидше на 2 секунди за найближчого аналога при виконані однакового коду експерименту, що не містить стороніх викликів, що приблизно відповідає 30 %.

Підсумовуючи, проведене тестування підтвердило високий рівень функціональності та продуктивності всіх компонентів розробленої технології

для проведення A/B тестування. У результаті виявлено, що система працює коректно та відповідає усім поставленим вимогам. Також було підтверджено, що швидкодія системи перевищує аналогічні рішення на ринку, забезпечуючи ефективну роботу та комфортне використання платформи для користувачів.

### 3.5 Висновок до розділу 3

Отже, всі компоненти інформаційної технології були успішно розроблені та реалізовані відповідно до вимог, визначених під час моделювання.

Отриманий програмний продукт відзначається високою швидкодією, повною функціональністю та зручністю в користуванні. Це відповідає усім критеріям задоволення інформаційних потреб користувачів, особливо при проведенні автоматизованого тестування.

Також, порівняння мов програмування TypeScript, C#, та Kotlin підтвердило їхню ефективність та придатність для створення інформаційної технології.

Інтегровані середовища розробки, такі як Microsoft Visual Studio, JetBrains Rider та Visual Studio Code, були порівнянні для подальшого забезпечення зручності та продуктивності при розробці та тестуванні програмного забезпечення.

# 4 ЕКОНОМІЧНА ЧАСТИНА

## 4.1 Оцінювання комерційного потенціалу розробки

Метою проведення комерційного та технологічного аудиту є оцінка можливостей для впровадження WEB-платформи, яка дозволить проводити A/B експерименти на конкретному WEB-сайті та максимізувати її комерційний потенціал.

Для проведення технологічного аудиту було залучено 3-х незалежних експертів Вінницького національного технічного університету д.т.н., проф. Яровий А.А., к.т.н., доцента Козловський А.В. з кафедри комп'ютерних наук та кафедри комп'ютерних систем управління д.т.н., проефсора Юхимчук М.С. Технологічний аудит проведемо з використанням таблиці 4.1 [20] де за п'ятибальною шкалою використовуючи 12 критеріїв оцінемо комерційний потенціал.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

| Критерії оцінювання та бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| Кри-терій | 0 | 1 | 2 | 3 | 4 |
| Технічна здійсненність концепції: | | | | | |
| 1 | Достовірність концепції не підтверджена | Концепція підтверджена експертними висновками | Концепція підтверджена розрахунками | Концепція перевірена на практиці | Перевірено роботоздатність продукту в реальних умовах |
| Ринкові переваги (недоліки): | | | | | |
| 2 | Багато аналогів на малому ринку | Мало аналогів на малому ринку | Кілька аналогів на великому ринку | Один аналог на великому ринку | Продукт не має аналогів на великому ринку |
| 3 | Ціна продукту значно вища за ціни аналогів | Ціна продукту дещо вища за ціни аналогів | Ціна продукту приблизно дорівнює цінам аналогів | Ціна продукту дещо нижче за ціни аналогів | Ціна продукту значно нижче за ціни аналогів |
| 4 | Технічні та споживчі властивості продукту значно гірші, ніж в аналогів | Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів | Технічні та споживчі властивості продукту на рівні аналогів | Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів | Технічні та споживчі властивості продукту значно кращі, ніж в аналогів |

Продовження табл. 4.1

| | | | | | |
|---|---|---|---|---|---|
| 5 | Експлуатаційні витрати значно вищі, ніж в аналогів | Експлуатаційні витрати дещо вищі, ніж в аналогів | Експлуатаційні витрати на рівні експлуатаційних витрат аналогів | Експлуатаційні витрати трохи нижчі, ніж в аналогів | Експлуатаційні витрати значно нижчі, ніж в аналогів |
| Ринкові перспективи | | | | | |
| 6 | Ринок малий і не має позитивної динаміки | Ринок малий, але має позитивну динаміку | Середній ринок з позитивною динамікою | Великий стабільний ринок | Великий ринок з позитивною динамікою |
| 7 | Активна конкуренція великих компаній на ринку | Активна конкуренція | Помірна конкуренція | Незначна конкуренція | Конкурентів немає |
| Практична здійсненність | | | | | |
| 8 | Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї | Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців | Необхідне незначне навчання фахівців та збільшення їх штату | Необхідне незначне навчання фахівців | Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї |
| 9 | Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні | Потрібні незначні фінансові ресурси. Джерела фінансування відсутні | Потрібні значні фінансові ресурси. Джерела фінансування є | Потрібні незначні фінансові ресурси. Джерела фінансування є | Не потребує додаткового фінансування |
| 10 | Необхідна розробка нових матеріалів | Потрібні матеріали, що використовуються у військово-промисловому комплексі | Потрібні дорогі матеріали | Потрібні досяжні та дешеві матеріали | Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві |
| 11 | Термін реалізації ідеї більший за 10 років | Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років | Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років |
| 12 | Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію | Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу | Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу | Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту | Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту |

Таблиця 4.2 – Рівні комерційного потенціалу розробки

| Середньоарифметична сума балів СБ, розрахована на основі висновків експертів | Рівень комерційного потенціалу розробки |
|---|---|
| 0-10 | Низький |
| 11-20 | Нижче середнього |
| 21-30 | Середній |
| 31-40 | Вище середнього |
| 41-48 | Високий |

У таблиці 4.3 містяться результати оцінки експертами можливостей розробки в аспекті комерційного потенціалу.

Таблиця 4.3 – Результати оцінювання комерційного потенціалу розробки

| Критерії | Прізвище, ініціали, посада експерта | | |
|---|---|---|---|
| | 1. Козловський А.В. | 2. Юхимчук М.С. | 3. Яровий А.А. |
| | Бали, виставлені експертами: | | |
| 1 | 3 | 4 | 3 |
| 2 | 2 | 4 | 4 |
| 3 | 4 | 4 | 4 |
| 4 | 3 | 4 | 3 |
| 5 | 3 | 3 | 3 |
| 6 | 4 | 4 | 4 |
| 7 | 4 | 2 | 2 |
| 8 | 3 | 4 | 4 |
| 9 | 3 | 3 | 3 |
| 10 | 4 | 4 | 4 |
| 11 | 4 | 3 | 4 |
| 12 | 4 | 4 | 4 |
| Сума балів | $СБ_1$=41 | $СБ_2$=43 | $СБ_3$=42 |
| Середньоарифметична сума балів $\overline{СБ}$ | (41+43+42) / 3 = 42 | | |

Середньоарифметична сума балів, розрахована на основі висновків експертів склала 42 бали, що згідно таблиці 4.2 вважається, що рівень комерційного потенціалу проведених досліджень є високий.

Ця розробка може бути комерціалізована через B2B модель продажів, призначену для власників WEB-сайтів, які мають на меті підвищити ефективність свого WEB-сайту шляхом зміни контенту та дизайну. Використовуючи цю платформу, клієнти мають можливість створювати експерименти, що дозволяють оцінити вплив нових змін на користувачів. Цей результат досягається за допомогою розподілу користувачів на дві групи: одна група бачить старий дизайн, а інша - новий. Після збору даних можна провести аналіз, щоб визначити, чи є нові зміни кращими за попередні.

Для впровадження цієї системи необхідно розгорнути розробку на серверах компанії та здійснити конфігураційні зміни, які будуть пов'язувати основний сайт компанії з розробленою платформою, відповідальною за виконання необхідних експериментів. Це дозволить ефективно впровадити технологічну інновацію та забезпечити максимальну користь для клієнтів.

В якості аналогу для даної розробки була вибрана платформа Adobe Target. Але варто відзначити, що цей аналог має свої недоліки. Зокрема, він не надає оновлення метрик у реальному часі, обмежує можливість встановлення додаткових умов для користувачів, які вже брали участь у попередніх експериментах, і не дозволяє розгортати систему на власних серверах.

У нашій розробці ці недоліки вирішуються шляхом розширення функціональності та підтримки швидкого розгортання системи. Крім того, наша система видається більш перспективною в порівнянні з аналогом за такими параметрами, як швидкодія та швидкість розгортання, завдяки використанню сучасних технологій без прив'язки до певної платформи.

Здійснимо аналіз якості та конкурентоспроможності нашої нової розробки порівняно з аналогом. Для цього використаємо таблицю 4.4, де представлені ключові техніко-економічні показники як аналогу, так і нової розробки.

Таблиця 4.4 – Основні параметри нової розробки та товару-конкурента

| Показник | Варіанти | | Відносний показник якості | Коефіцієнт вагомості параметра |
|---|---|---|---|---|
| | Базовий (товар-конкурент) | Новий (інноваційне рішення) | | |
| 1 | 2 | 3 | 4 | 5 |
| Час розгортання змін, мс | 20 | 100 | 5 | 25% |
| Кількість паралельних запитів | 1 | 4 | 4 | 25% |
| Максимальна кількість активних користувачів | 40 | 80 | 2 | 10% |
| Обсяг використаних ресурсів, % | 70 | 90 | 1,3 | 20% |
| Час виконання запитів, мс | 500 | 150 | 3,3 | 20% |

Проведемо оцінку якості продукції, яка є найефективнішим засобом забезпечення вимог споживачів та порівняємо її з аналогом.

Визначимо відносні одиничні показники якості по кожному параметру за формулами (4.1) та (4.2) і занесемо їх у відповідну колонку табл. 4.5.

$$q_i = \frac{P_{Hi}}{P_{Бi}}, \tag{4.1}$$

або

$$q_i = \frac{P_{Бi}}{P_{Hi}}, \tag{4.2}$$

де $P_{Hi}$, $P_{Бi}$ – числові значення $i$-го параметру відповідно нового і базового виробів.

$$q_1 = \frac{100}{20} = 5;$$

$$q_2 = \frac{4}{1} = 4;$$

$$q_3 = \frac{80}{40} = 2;$$

$$q_4 = \frac{90}{70} = 1{,}3;$$

$$q_5 = \frac{500}{150} = 3{,}3.$$

Відносний рівень якості нової розробки визначаємо за формулою:

$$К_{я.в.} = \sum_{i=1}^{n} q_i \cdot \alpha_i , \qquad (4.3)$$

$$К_{я.в.} = 5 \cdot 0{,}25 + 4 \cdot 0{,}25 + 2 \cdot 0{,}1 + 1{,}3 \cdot 0{,}2 + 3{,}3 \cdot 0{,}2 = 3{,}37.$$

Відносний коефіцієнт показника якості нової розробки більший одиниці, отже нова розробка якісніший базового товару-конкурента.

Наступним етапом є визначення конкурентоспроможності продукту. Конкурентоспроможність продукту є ключовою умовою успішності підприємства на ринку та важливою основою для отримання прибутку від його діяльності.

Однією з умов вибору продукту споживачем є відповідність основних ринкових характеристик продукту умовним характеристикам конкретних потреб покупця. Цими характеристиками зазвичай є стандарти та технічні параметри, а також вартість придбання та витрати на використання продукту.

В таблиці 4.5 наведено технічні та економічні показники для розрахунку конкурентоспроможності нової розробки відносно товару-аналога, технічні дані взяті з попередніх розрахунків.

Таблиця 4.5 – Нормативні, технічні та економічні параметри нової розробки і товару-виробника

| Показники | Варіанти | |
|---|---|---|
| | Базовий (товар-конкурент) | Новий (інноваційне рішення) |
| 1 | 2 | 3 |
| 1. Нормативно-технічні показники | | |
| Час розгортання змін, мс | 20 | 100 |
| Кількість паралельних запитів | 1 | 4 |
| Максимальна кількість активних користувачів | 40 | 80 |
| Обсяг використаних ресурсів, % | 70 | 90 |
| Час виконання запитів, мс | 500 | 150 |
| 2. Економічні показники | | |
| Ціна придбання, грн. | 15000 | 10000 |

Загальний показник конкурентоспроможності інноваційного рішення (К) з урахуванням вищезазначених груп показників можна визначити за формулою:

$$K = \frac{I_{m.n.}}{I_{e.n.}}, \tag{4.4}$$

де $I_{m.n.}$ – індекс технічних параметрів; $I_{e.n.}$ – індекс економічних параметрів.

Індекс технічних параметрів є відносним рівнем якості інноваційного рішення. Індекс економічних параметрів визначається за формулою (4.5)

$$I_{e.n.} = \frac{\sum_{i=1}^{n} P_{Hei}}{\sum_{i=1}^{n} P_{Бei}}, \tag{4.5}$$

де $P_{Hei}$, $P_{Бei}$ – економічні параметри (ціна придбання та споживання товару) відповідно нового та базового товарів.

$$I_{е.п.} = \frac{10000}{15000} = 0,66;$$

$$К = \frac{3,37}{0,66} = 5,1.$$

Зважаючи на розрахунки, можна зробити висновок, що нова розробка буде конкурентоспроможніше, ніж конкурентний товар.

## 4.2 Прогнозування витрат на виконання науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи, розподіляються на кілька категорій: витрати на оплату праці, витрати на соціальні заходи, витрати на матеріали, витрати на паливо та енергію для наукових цілей, витрати на службові відрядження, витрати на програмне забезпечення для наукових досліджень, інші видатки та загальні накладні витрати.

1. Основна заробітна плата кожного із дослідників $З_О$, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_0 = \frac{М}{Т_р} * t \text{ (грн)}, \quad (4.6)$$

де $M$ – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$T_P$ – число робочих днів в місяці; приблизно $T_P \approx 21...23$ дні;

$t$ – число робочих днів роботи дослідника.

Для створення WEB-платформи, яка забезпечить можливість проведення A/B тестів на певному сайті, потрібно найняти програміста з щомісячним окладом у розмірі 10000 гривень. Із 23 робочих днів на місяць, програміст буде працювати 19 днів. Підсумкові розрахунки наведено в таблиці 4.6 для подальшого аналізу.

Таблиця 4.6 – Заробітна плата дослідника в науковій установі бюджетної сфери

| Найменування посади | Місячний посадовий оклад, грн. | Оплата за робочий день, грн. | Число днів роботи | Витрати на заробітну плату грн. |
|---|---|---|---|---|
| Керівник | 11000 | 478,26 | 19 | 9086,94 |
| Інженер-програміст | 10000 | 434,78 | 19 | 8260,86 |
| Всього | | | | 17347,7 |

2. Розрахунок додаткової заробітної плати робітників

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які приймали устають в розробці нового технічного рішення розраховується як 10 - 12 % від основної заробітної плати робітників.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати.

$$З_д = (З_о + З_р) * \frac{Н_{доп}}{100\%}, \quad (4.7)$$

Зд = (17347,7 · 12 % / 100 % ) = 2081,64 (грн).

3. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою (4.3):

$$Н_{ЗП} = (З_о + З_д) * \frac{\beta}{100} \text{ (грн)}, \quad (4.8)$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.;

$\beta$ – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % .

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді:

Нз = (17347,7+ 2081,64) · 22 % / 100 % = 4274,45 (грн).

4. Витрати на матеріали М та комлектуючі К, що були використані під час виконання даного етапу роботи, розраховуються по кожному виду матеріалів за формулою:

$$М = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \text{ грн.}, \quad (4.9)$$

де $Н_i$ – витрати матеріалу *і*-го найменування, кг;

$Ц_i$ – вартість матеріалу *і*-го найменування, грн./кг.;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

$В_i$ – маса відходів матеріалу *і*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *і*-го найменування, грн/кг;

n – кількість видів матеріалів.

Витрати на комплектуючі вироби, які використовують при виготовленні одиниці продукції, розраховуються, згідно їх номенклатури, за формулою:

$$К=\sum_{i=1}^{n} Н_i \cdot Ц_i \cdot К_i, \quad (4.10)$$

де $Н_i$ – кількість комплектуючих *i*-го виду, шт.;

$Ц_i$ – покупна ціна комплектуючих *i*-го найменування, грн.;

$К_i$ – коефіцієнт транспортних витрат (1,1…1,15).

Інформацію про використані матеріали та комлектуючі подамо у вигляді табл. 4.7.

Таблиця 4.7 – Матеріали, що використані на розробку

| Найменування матеріалу | Ціна за одиницю, грн. | Витрачено | Вартість витраченого матеріалу, грн. |
|---|---|---|---|
| Папір | 200 | 1 | 200 |
| Ручка | 15 | 1 | 15 |
| CD-диск | 12 | 1 | 12 |
| Флешка | 300 | 1 | 300 |
| Всього | | | 527 |
| З врахуванням коефіцієнта транспортування | | | 579,7 |

5. Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження.

Балансову вартість програмного забезпечення розраховують за формулою:

$$В_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot С_{пргі} \cdot К_i, \quad (4.11)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу цього виду, грн;

$С_{пргі}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

$К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо ($К_і$ = 1,10…1,12).

k – кількість найменувань програмних засобів.

Отримані результати занесемо в таблицю 4.8.

Таблиця 4.8 – Витрати на придбання програмних засобів по кожному виду

| Найменування устаткування | Кількість, шт | Ціна за одиницю, грн | Вартість |
|---|---|---|---|
| Ключ для активації середовища Visual Studio | 1 | 1500 | 1500 |
| Docker license | 1 | 5500 | 5500 |
| Всього | | | 7000 |

6. Амортизація обладнання, комп’ютерів та приміщень, які використовувались під час виконання даного етапу роботи

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

$$А = \frac{Ц * Т}{12 * Т_{кор.}}, \quad (4.12)$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн.;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункта 137.3.3 Податкового кодекса амортизація нараховується на основні засоби вартістю понад 2500 грн. В нашому випадку для написання

магістерської роботи використовувався персональний комп'ютер вартістю 10000 грн.

$$A = \frac{10000 \cdot 1}{2 \cdot 12} = 416{,}67$$

7. До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на всі види палива й енергії, що безпосередньо використовуються з технологічною метою на проведення досліджень.

$$B_e = \sum_{i=1}^{n} \frac{W_{yt} \cdot t_i \cdot Ц_e \cdot К_{впі}}{\eta_i}, \qquad (4.13)$$

де $W_{yt}$ – встановлена потужність обладнання на певному етапі розробки, кВт;

$t_i$ – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн;

$К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі} < 1$;

$\eta_i$ – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

Для написання магістерської роботи використовується персональний комп'ютер для якого розрахуємо витрати на електроенергію.

$$Be = \frac{0{,}3 * 160 * 7{,}5 * 0{,}5}{0{,}8} = 225$$

Накладні (загальновиробничі) витрати Внзв охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо. Накладні (загальновиробничі) витрати Внзв можна прийняти як (100…150)% від суми

основної заробітної плати розробників та робітників, які виконували дану МКНР, тобто:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%},$$

(4.14)

де $Н_{нзв}$ – норма нарахування за статтею «Інші витрати».

Внзв = 17347,7 * 100/100% = 17347,7 грн.

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКНР

В = 17347,7+4274,45+4274,45+579,7+7000+416,67+225+ 17347,7=51465.67 грн.

Прогнозування загальних втрат ЗВ на виконання та впровадження результатів виконаної МКНР здійснюється за формулою:

$$ЗВ = \frac{В}{\eta},$$ (4.15)

де η – коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії науково-дослідних робіт, то коефіцієнт $\beta$ = 0,9.

Звідси:

ЗВ = 51465.67/0.9 =57184.07 грн.

## 4.3 Розрахунок економічної ефективності науково-технічної розробки

В цьому розділі планується здійснити кількісний прогноз, щодо очікуваної користі та прибутку, які можна отримати в майбутньому завдяки впровадженню результатів проведених наукових досліджень. Розглянемо, як змінюється чистий прибуток підприємства (ΔПі) протягом років, коли очікується отримання позитивних результатів від впровадження розробки, за допомогою наступної формули:

$$\Delta П_i = \sum_{1}^{n} \left( \Delta Ц_o \cdot N + Ц_o \cdot \Delta N \right)_i \cdot \lambda \cdot \rho \cdot \left( 1 - \frac{\upsilon}{100} \right), \quad (4.16)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році.

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки:

Цо – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки:

л – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт *л* = 0,8333.

р – коефіцієнт, який враховує рентабельність продукту. р = 0,25;

х – ставка податку на прибуток. У 2023 році – 18%.

Припустимо, що внаслідок впровадження наукової розробки вдосконалюється якість продукції, що дозволяє збільшити її ціну на 1000 гривень. Також передбачається збільшення кількості продукції, яка буде

реалізована: протягом першого року на 22 одиниці, протягом другого року – на 18 одиниць, протягом третього року – на 12 одиниць. Перед впровадженням розробки було продано 150 одиниць продукції по ціні 10000 гривень. Розрахуємо прибуток, який отримає підприємство протягом трьох років.

$$\Delta П_1 = [1000 \cdot 150 + (10000 + 1000) \cdot 22] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 66963{,}98 \text{ грн.}$$

$$\Delta П_2 = [1000 \cdot 150 + (10000 + 1000) \cdot (22 + 18)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 225163{,}66 \text{ грн.}$$

$$\Delta П_3 = [1000 \cdot 150 + (10000 + 1000) \cdot (22 + 18 + 12)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 247712{,}76 \text{ грн.}$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки.

$$PV = K_{інв} * ЗВ, \quad (4.17)$$

$k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо ($k_{інв}$ = 2…5).

PV = 2 * 57184.07 = 114368.14.

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули:

$$\text{Е}_{\text{абс}} = (\text{ПП} - \text{PV}), \tag{4.18}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн.;

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \tag{4.19}$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДЦКР, грн.;

Т – період часу, протягом якою виявляються результати впровадженої НДДКР, роки;

$\tau$ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,2;

t – період часу (в роках).

$$\text{ПП} = \frac{66963,99}{(1+0,2)^1} + \frac{225163,66}{(1+0,2)^2} + \frac{247712,76}{(1+0,2)^3} = 356186,02 \text{ грн.}$$

Еабс ( 356186,02 – 114368.14) =241817.88 грн.

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього користуються формулою:

$$E_в = \sqrt[Тж]{1 + \frac{E_{abc}}{PV}} - 1, \quad (4.20)$$

$T_{ж}$ – життєвий цикл наукової розробки, роки.

Ев = 3√(1+(241817.88/114368.14)) – 1 = 0.46 = 46%.

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \quad (4.21)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d= (0,17…0,21);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1).

Tmin = 0,21+0,05 = 0,215.

Так як $E_в > \tau_{min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$Т_{ок} = \frac{1}{E_в}, \quad (4.22)$$

Ток = 1/0.46 = 2.17 роки.

Так як $T_{ок} \leq$ 3...5-ти років, то фінансування даної наукової розробки в принципі є доцільним.

## 4.5 Висновок до розділу 4

Була проведена оцінка комерційного потенціалу розробки WEB-платформи, яка дозволяє керувати A/B експериментами на певному сайті і перевершує середній рівень якості. Порівняльний аналіз з аналогічним продуктом показав, що нова розробка є вищою за якістю та конкурентоспроможністю, а також вигіднішою з технічних і економічних позицій. Прогнозовані витрати на науково-дослідну роботу для кожної з частин проекту складатимуть 51,187.05 грн. Загальні витрати на виконання та впровадження результатів даного дослідження становитимуть 56,874.5 грн. Вкладені інвестиції в цей проект повернуться через 2 роки за очікуваної прибутковості у сумі 356186,02 грн протягом трьох років.

# ВИСНОВКИ

Під час виконання магістерської роботи було розглянуто питання актуальності розробки інформаційної технології A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей.

У ході роботи отримав подальшого розвитку метод проведення A/B тестування, а саме здійснено опис та реалізацію поєднання A/B тестування з використанням великих мовних моделей.

В рамках дослідження було досягнуто значного підвищення релевантності контенту WEB-сайтів, що стало можливим завдяки інноваційній інтеграції з великими мовними моделями. Цей підхід виявився унікальним і відрізняється від традиційних систем, які використовуються в інших A/B тестуваннях. Одним з ключових аспектів цієї інтеграції є здатність системи розпізнавати та враховувати унікальні характеристики кожного користувача, такі як, наприклад, географічне місцезнаходження.

Отримані результати свідчать про значні переваги інтеграції великих мовних моделей у процес A/B тестування, відкриваючи нові можливості для розвитку та оптимізації WEB-сайтів. Така інтеграція не тільки підвищує релевантність контенту, але й вносить інновації в спосіб взаємодії з користувачами, що ставить цю систему на передовій позиції у сфері WEB-розробки та цифрового маркетингу.

В ході експериментального тестування програмного додатку було продемонстровано, що система здатна автоматично налаштовувати контент на основі індивідуальних особливостей користувачів. Зокрема, використання IP-адреси для визначення мови користувача дозволило автоматично адаптувати контент сайту до мови відвідувача. Це не лише підвищує релевантність контенту, але й значно покращує загальний користувацький досвід, забезпечуючи більш персоналізований та цільовий підхід.

На основі аналізу стану питання сучасного розвитку технологій проведення A/B тестування доведено актуальність розробки. Проведено

аналіз методів та засобів вирішення задачі проведення A/B тестувань. Розглянуто аналоги технології та на основі їх порівняння встановлено вимоги та завдання для власної розробки.

В процесі моделювання визначено, що інформаційні технологія буде складатися з клієнтської та серверної частини. На основі аналізу було побудовано структурну схему системи. Було описано можливість інтеграції з великим мовними моделями для підтримки високого рівня адаптації WEB-сайтів. Також, було здійснено проектування бази даних, в результаті чого було слкадено універсальне відношення.

Було розглянуто сучасні мови програмування для розробки даних систем, на основі чого було обрано C# мовою для розробки даного додатку у середовищі  Visual Studio. Також, було розглянуто важливість проведення тестувань додатку, та було використано Unit тестування фукнціональності.

Важливим аспектом роботи було тестування швидкодії розробленої платформи. Результати тестування показали, що наша платформа працює швидше на 2 секунди порівняно з найближчим аналогом при виконанні однакового коду експерименту, що не містить сторонніх викликів. Це становить приблизно 30% покращення швидкодії, що є значним досягненням у плані ефективності системи.

Виконано економічний аналіз та розрахунки, результати яких підтвердили економічну доцільність даної розробки.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бугайов В. Ю., Козловський А. В., «Аналіз сучасного рівня розвитку інформаційних технологій A/B тестування», Матеріали IV Всеукраїнської студентської наукової конференції «Розвиток сучасної науки: актуальні питання теорії та практики» (2023). URL: https://archive.liga.science/index.php/conference-proceedings/issue/view/ukr-17.11.2023 (дата звернення 18.11.2023).

2. Бугайов В. Ю., Козловський А. В., «Методи вирішення задачі адаптації WEB-сайту з використанням A/B тестування та великих мовних моделей», Матеріали IV Міжнародна наукова конференція «Розвиток наукової думки постіндустріального суспільства: сучасний дискурс» (2023) URL: https://archive.mcnd.org.ua/index.php/conference- proceeding/issue/view/ 17.11.2023 (дата звернення 19.11.2023).

3. What is A/B testing? And why is it so important? URL: https://www.fullstory.com/ab-testing/ (дата звернення 09.10.2023).

4. What Are the Benefits of A/B Testing? URL: https://devcycle.com/blog/what-are-the-benefits-of-a-b-testing (дата звернення 09.10.2023).

5. A Refresher on A/B Testing. URL: https://hbr.org/2017/06/a-refresher-on-ab-testing (дата звернення 12.10.2023).

6. What is A/B testing? URL: https://www.optimizely.com/optimization-glossary/ab-testing/ (дата звернення 13.10.2023).

7. What is a large language model (LLM)? URL: https://www.cloudflare.com/learning/ai/what-is-large-language-model/ (дата звернення 15.10.2023).

8. Google Optimize. URL: https://en.wikipedia.org/wiki/Google_Optimize (дата звернення 16.10.2023).

9. Introduction to Target. URL: https://experienceleague.adobe.com/docs /target/using/introduction/intro.html (дата звернення 18.10.2023).

10. Crazy Egg. URL: https://www.crazyegg.com/ (дата звернення 20.10.2023).

11. How to Do A/B Testing: 15 Steps for the Perfect Split Test. URL: https://blog.hubspot.com/marketing/how-to-do-a-b-testing (дата звернення 09.11.2023).

12. Мартін Р. Чистий код: створення і рефакторинг за допомогою Agile – Харків: Видавництво «Фабула», 2019 – 98 с

13. Троельсен А. C# 6.0 and the .NET 4.6 Framework – Нью-Йорк: Apress, 2015 – 1018 с.

14. Database Structure and Design Tutorial. URL: https://www.lucidchart.com/pages/database-diagram/database-design (дата звернення 10.11.2023).

15. Algorithm Design. URL: https://reintech.io/terms/category/algorithm-design-in-software-development (дата звернення 13.11.2023).

16. Machine code (machine language). URL: https://www.techtarget.com/whatis/definition/machine-code-machine-language (дата звернення 15.10.2023).

17. Ріхтер Д. CLR via C#. – Редмонд, Вашингтон: Microsoft Press, 2012. – 214

18. Троельсен А. Pro C# 7: With .NET and .NET Core – Нью-Йорк: Apress, 2017 – 618 с.

19. Write your first analyzer and code fix. URL: https://docs.microsoft.com/en-us/dotnet/csharp/roslyn-sdk/tutorials/how-to-write-csharp-analyzer-code-fix (дата звернення 20.10.2023).

20. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

# ДОДАТКИ

**Додаток А (обов'язковий) Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень**

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія A/B тестування для адаптації WEB-сайту з використанням великих мовних моделей

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

**Показники звіту подібності Unicheck**

Оригінальність 95,47% Схожість 4,53%

**Аналіз звіту подібності** (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Бугайов В.Ю.

Керівник роботи Козловський А.В.

**Опис прийнятого рішення**

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

## Додаток Б (обов’язковий)

## Лістинг програми

ListOfExperiments.tsx

```
import { Box, Button, Divider, List, ListItem, ListItemButton, ListItemIcon, ListItemText, TextField } from '@mui/material'
import React, { Fragment, useEffect, useState } from 'react'
import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
import './CreateExperimentStyle.css';
const axios = require('axios').default;

interface Experiment{
  id:number;
  name: string;
  description: string;
  assignment: string;
}

export const ListOfExperiments: React.FC = () => {
  const [listOfExperiments, setListOfExperiments] = useState([]);

  async function getExperiments() {
    try {
        const response = await axios.get(BACK_END_HOST_LOCATION + '/api/experiment', axiosConfig);
      setListOfExperiments(response?.data?.map(experiment =>  {
        return <Fragment>
       <ListItem disablePadding>
                                                <ListItemButton component="a" href={`/experiment/${experiment.experimentId}`}>
         <ListItemText primary={experiment.experimentId} />
         <ListItemText primary={experiment.name} />
       </ListItemButton>
     </ListItem>
           <Divider />
    </Fragment>
    }
  ))
    } catch (error) {
      alert(error);
    }
   }
```

```
  useEffect(() => {
     getExperiments();
    },[]);
 return (
  <Fragment>
      <Box sx={{ width: '70%', bgcolor: 'background.paper' }}>
     <nav aria-label="main mailbox folders">
       <List>
       <Divider />
       {listOfExperiments}
       </List>
       </nav>
       </Box>
  </Fragment>
 )
}
```

ViewExperiment.tsx

```
import { Box, Button, Dialog, DialogActions, DialogContent, DialogContentText,
DialogTitle, Divider, Fab, List, ListItem, ListItemButton, ListItemText, TextField
} from '@mui/material'
import React, { Fragment, useEffect, useState } from 'react'
import { Redirect, useHistory, useParams } from 'react-router-dom';
import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
import './ViewExperimentStyle.css';
import AddIcon from '@mui/icons-material/Add';
const axios = require('axios').default;
import DeleteForeverIcon from '@mui/icons-material/DeleteForever';

export const ViewExperiment: React.FC = () => {
  let { id } = useParams<{ id?: string }>();
  let { variationId } = useParams<{ variationId?: string }>();
  const [listOfVariations, setListOfVariations] = useState([]);
  const [listOfVariationsData, setListOfVariationsData] = useState([]);
  const [variationName, setVariationName] = useState("");
  const [variationCode, setVariationCode] = useState("");
  const [trafficPercentage, setTrafficPercentage] = useState([]);
  const [open, setOpen] = React.useState(false);
  const [isEnabled, setIsEnabled] = useState(false);

    const handleClickOpen = () => {
      setOpen(true);
    };
```

```
    const handleClose = () => {
     history.push(`/experiment/${id}`);
      setOpen(false);
    };

    useEffect(() => {
     loadExperiment();
    },[]);

    async function deleteVariation(event, variationId1){
     try {
        const response = await axios.delete(BACK_END_HOST_LOCATION +
`/api/variation/${variationId1}`, axiosConfig);
      history.go(0)
     } catch (error) {
      alert(error);
    }
   }

   const [name, setName] = React.useState('');

   const handleNameChange = (event) => {
     setName(event.target.value);
   };
   const handleVariationCodeChange = (event) => {
    setVariationCode(event.target.value);
  };
  const handleVariationNameChange = (event) => {
   setVariationName(event.target.value);
 };

   const [description, setDescription] = React.useState('');

   const handleDescriptionChange = (event) => {
     setDescription(event.target.value);
   };

   const [assignment, setAssignment] = React.useState('');

   const handleAssignmentChange = (event) => {
     setAssignment(event.target.value);
   };


  async function loadExperiment() {
```

```
    try {
        const response = await axios.get(BACK_END_HOST_LOCATION + `/api/experiment/${id}`, axiosConfig);
      setIsEnabled(response?.data?.isEnabled);
      setName(response.data.name);
      setDescription(response?.data?.description);
      setAssignment(response?.data?.assignment)
      if(variationId){
       if(response?.data?.variations.find(v=>v.variationId == variationId)?.name)
            setVariationName(response?.data?.variations.find(v=>v.variationId == variationId)?.name);
                        if(response?.data?.variations.find(v=>v.variationId == variationId)?.executionCode)
             setVariationCode(response?.data?.variations.find(v=>v.variationId == variationId)?.executionCode);
       handleClickOpen();
      }
      setListOfVariationsData(response?.data?.variations);
      var variations = response?.data?.variations.map(v=>{
                                    return {variationId: v.variationId, percentageOfTraffic:v.percentageOfTraffic, name:v.name }});
      setTrafficPercentage(variations);
      setListOfVariations(response?.data?.variations.map(variation => {
        return <Fragment>
     <ListItem disablePadding>
                    <ListItemButton sx={{width:"160px"}} component="a" href={`/experiment/${id}/${variation.variationId}`}>
       <ListItemText primary={variation.variationId} />
       <ListItemText primary={variation.name} />
       </ListItemButton>
     <ListItemButton onClick={e=>deleteVariation(e, variation.variationId)}>
     <DeleteForeverIcon />
     </ListItemButton>
    </ListItem>
          <Divider />
   </Fragment>
   }
  ))
    } catch (error) {
     alert(error);
    }
   }

  const onCreateButtonClick = () => {
    updateExperiment();
```

```
  }

  async function updateExperiment() {
   try {
          const data = { name: name, description: description, assignment: assignment};
        const response = await axios.put(BACK_END_HOST_LOCATION + `/api/experiment/${id}`, data, axiosConfig);
    } catch (error) {
     alert(error);
    }
  }
  let history = useHistory();

  async function addVariation(){
      const data = { name: 'New Variation', executionCode: '', experimentId: Number.parseInt(id)};
       const response = await axios.post(BACK_END_HOST_LOCATION + `/api/variation`, data, axiosConfig);

    setVariationName('New Variation');
    setVariationCode('');
    history.push(`/experiment/${id}/${response.data}`);
    handleClickOpen();
  }

  async function handleUpdateVariation(){
   try {
     const data = { name: variationName, executionCode: variationCode};
                    await axios.put(BACK_END_HOST_LOCATION + `/api/variation/${variationId}`, data, axiosConfig);
     handleClose();
     history.push(`/experiment/${id}`);
     history.go(0)
    } catch (error) {
     alert(error);
   }
  }

  function updateTraffic(id, traffic) {
   const prevState =  JSON.parse(JSON.stringify(trafficPercentage));
   prevState.find(v => v.variationId == id).percentageOfTraffic = traffic;
   setTrafficPercentage(prevState);
  }
```

```
  function getVariationsTraffic() {
    return trafficPercentage.map((v)=>{
      return (
        <div className="viewexperiment_variationtraffic">
        <label className="viewexperiment_label">
           {v.name}
         </label>
                                <input type="text" style={{width: "30PX"}}
defaultValue={v.percentageOfTraffic} onChange={th =>
updateTraffic(v.variationId, th.currentTarget.value)} />
         </div>)
    })
  }

  async function onChangeTrafficButtonClick(){
                                                var sum =
trafficPercentage.map(t=>Number.parseInt(t.percentageOfTraffic)).reduce((partial
_sum, a) => partial_sum + a, 0)
    if(sum!=100)
    alert("Сума варіацій не рівна 100%");
    else{
      var diff = [];
      trafficPercentage.forEach(t=>{
          if(t.percentageOfTraffic != listOfVariationsData.find(l=>l.variationId ==
t.variationId).percentageOfTraffic)
        {
                                diff.push({id: t.variationId, percentageOfTraffic:
Number.parseInt(t.percentageOfTraffic)});
        }
      })
      if(diff.length != 0 ){
        try {
                            await axios.put(BACK_END_HOST_LOCATION +
`/api/variation/updatetraffic`, diff, axiosConfig);
         history.go(0)
        } catch (error) {
         alert(error);
       }
      }
    }
  }

  async function changeStatus(){
    setIsEnabled(!isEnabled);
    try {
```

```
        const response = await axios.put(BACK_END_HOST_LOCATION +
`/api/experiment/changestatus/${id}`, axiosConfig);
      } catch (error) {
       alert(error);
      }
    }

  return (
   <Fragment>
     <div className="viewExperiment_root">
     <Dialog open={open} onClose={handleClose}>
      <DialogTitle>          Варіація {variationId}
</DialogTitle>
      <DialogContent>

        <TextField
         autoFocus
         margin="dense"
         id="name"
         label="Назва"
         type="email"
         fullWidth
         variant="standard" sx={{marginBottom:"55px"}}
         value={variationName}
         onChange={handleVariationNameChange}
        />
        <TextField
        id="outlined-multiline-static"
        label="Код виконання на сторінці"
        multiline
        rows={4}
        sx={{marginBottom: "20px", width: "65%"}}
        value={variationCode}
        onChange={handleVariationCodeChange}
       />
       </DialogContent>
       <DialogActions>
        <Button onClick={handleClose}>Закрити</Button>
        <Button onClick={handleUpdateVariation}>Оновити</Button>
       </DialogActions>
     </Dialog>
     <div className="viewExperiment_variations">
     <Box sx={{ bgcolor: 'background.paper' }}>
     <nav aria-label="main mailbox folders">
      <List>
```

```
      <Divider />
      {listOfVariations}
      <ListItem disablePadding>
            <Fab color="primary" aria-label="add" sx={{ height: "26px"}}
onClick={addVariation} >
 <AddIcon sx={{ height: "26px"}} />
</Fab>          <ListItemText primary="Створити варіацію" sx={{paddingLeft:
"19px"}} />
     </ListItem>

      </List>
      </nav>
      </Box>
    </div>
    <div className="viewExperiment_container">
          <TextField id="outlined-basic" label="Назва" variant="outlined"
sx={{marginBottom: "20px", width: "65%"}}
          value={name}
          onChange={handleNameChange} />
    <TextField
      id="outlined-multiline-static"
      label="Опис"
      multiline
      rows={4}
      sx={{marginBottom: "20px", width: "65%"}}
      value={description}
      onChange={handleDescriptionChange}
     />
        <TextField
      id="outlined-multiline-static"
      label="Умова попадання на експеримент (JavaScript)"
      multiline
      rows={4}
      sx={{marginBottom: "20px", width: "65%"}}
      value={assignment}
      onChange={handleAssignmentChange}
     />
       <div>
                                          <Button          variant="contained"
onClick={onCreateButtonClick}>Оновити</Button>
    </div>
    <div className="trafficPercantage_control">
    <div >
     {getVariationsTraffic()}
```

```
                    <Button sx={{marginTop: "20px"}} variant="contained" onClick={onChangeTrafficButtonClick}>Змінити розподіл трафіку</Button>
      </div>
  {!isEnabled?
        <Button variant="contained" color="success" sx={{height: "40px",width: "250px"}} onClick={changeStatus}>
  Запустити експеримент
</Button> : <Button variant="contained" color="error" sx={{height: "40px",width: "250px"}} onClick={changeStatus}>
  Зупинити експеримент
</Button>}
</div>
      </div>
      </div>
   </Fragment>
  )
}

ViewExperimentStyle.css

.viewExperiment_root{
  display: flex;
}

.viewExperiment_container{
  display: flex;
  padding-left: 100px;
  flex-direction: column;
  width: 98%;
}

.trafficPercantage_control{
  flex-direction: row;
  display: flex;
}

.viewExperiment_variations {
  padding-left: 27px;
}

.viewexperiment_variationtraffic {
  display: flex;
  flex-direction: row;
  padding-top:20px;
  width: 200px;
```

```
}
.viewexperiment_label{
  padding-right: 10px;
}
```

CreateExperiment.tsx

```
import { Button, TextField } from '@mui/material'
import React, { Fragment } from 'react'
import { useHistory } from 'react-router-dom';
import { axiosConfig, BACK_END_HOST_LOCATION } from '../Configuration';
import './CreateExperimentStyle.css';
const axios = require('axios').default;

export const CreateExperiment: React.FC = () => {
  let history = useHistory();

  async function createExperiment() {
    try {
          const data = { name: name, description: description, assignment: assignment};
        const response = await axios.post(BACK_END_HOST_LOCATION + '/api/experiment', data, axiosConfig);
      history.push(`/experiment/${response.data}`);
      history.go(0)
    } catch (error) {
      alert(error);
    }
   }

  const [name, setName] = React.useState('');

  const handleNameChange = (event) => {
    setName(event.target.value);
  };

  const [description, setDescription] = React.useState('');

  const handleDescriptionChange = (event) => {
    setDescription(event.target.value);
  };

  const [assignment, setAssignment] = React.useState('');

  const handleAssignmentChange = (event) => {
```

```
    setAssignment(event.target.value);
  };

  const onCreateButtonClick = () => {
    createExperiment();
   }

 return (
  <Fragment>
    <div className="createExperiment_container">
              <TextField id="outlined-basic" label="Назва" variant="outlined"
sx={{marginBottom: "20px", width: "65%"}}
              value={name}
              onChange={handleNameChange} />
    <TextField
      id="outlined-multiline-static"
      label="Опис"
      multiline
      rows={4}
      sx={{marginBottom: "20px", width: "65%"}}
      value={description}
      onChange={handleDescriptionChange}
     />
         <TextField
      id="outlined-multiline-static"
      label="Умова попадання на експеримент (JavaScript)"
      multiline
      rows={4}
      sx={{marginBottom: "20px", width: "65%"}}
      value={assignment}
      onChange={handleAssignmentChange}
     />
       <div>
                                              <Button variant="contained"
onClick={onCreateButtonClick}>Створити</Button>
    </div>
    </div>
  </Fragment>
 )
}
```

**Додаток В (обов'язковий)**

# ІЛЮСТРАТИВНА ЧАСТИНА

## ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ А/В ТЕСТУВАННЯ ДЛЯ АДАПТАЦІЇ WEB-САЙТУ З ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

Виконав: студент 2-го курсу,
групи 2КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Бугайов В.Ю.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. КН
_____ Козловський А.В.
(прізвище та ініціали)
«05» 12 _____ 2023 р.

Вінниця ВНТУ - 2023 рік

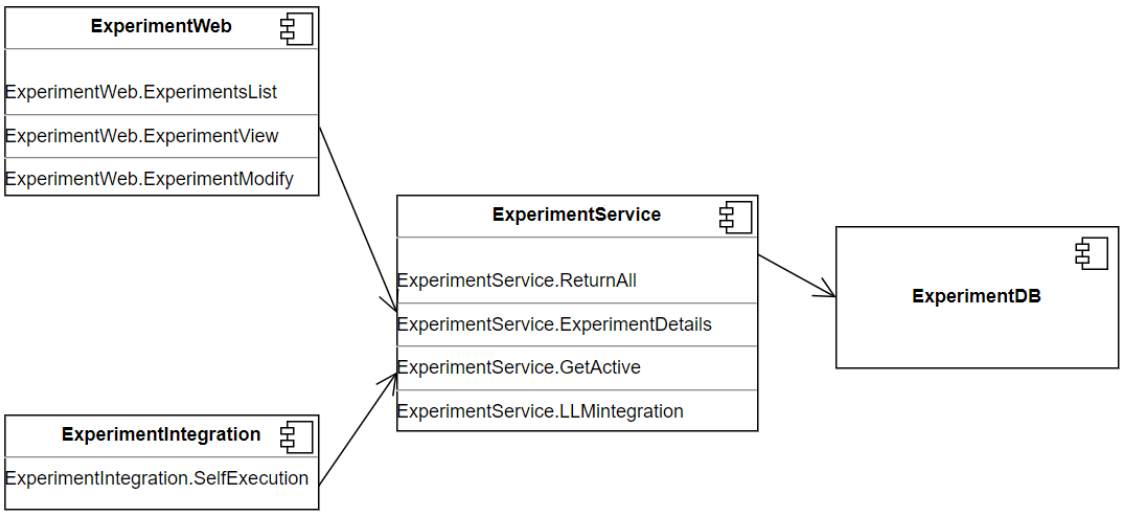



Рисунок В.1 – Структурна схема інформаційної технології

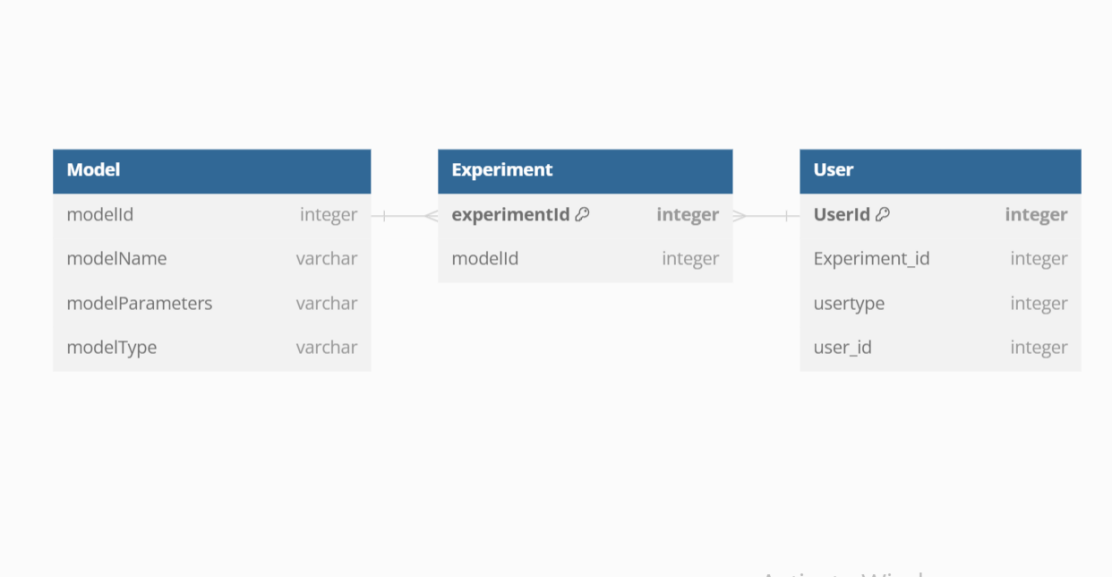



Рисунок В.2 – Структура бази даних

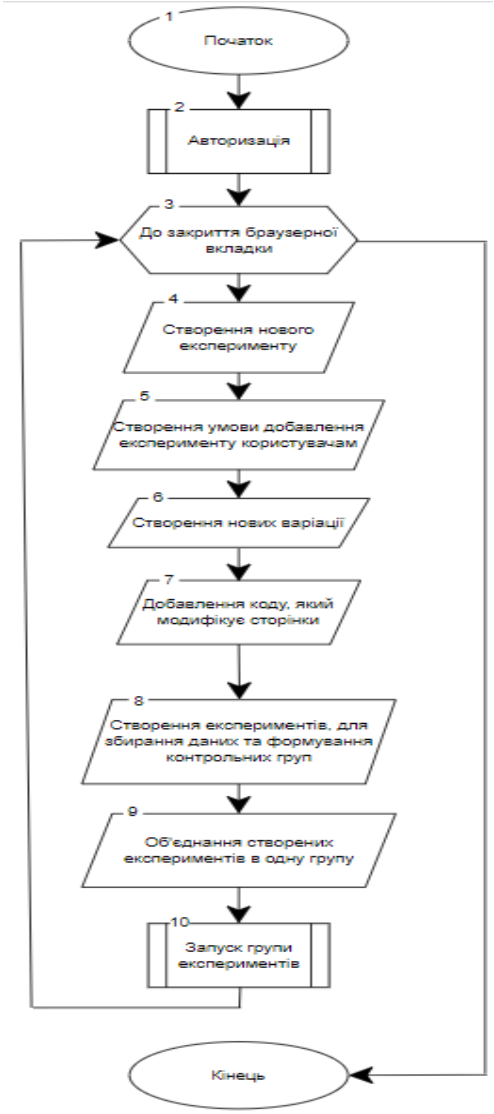



Рисунок В.3 – Алгоритм роботи інформаційної технології A/B тестування для адаптації WEB-сайту

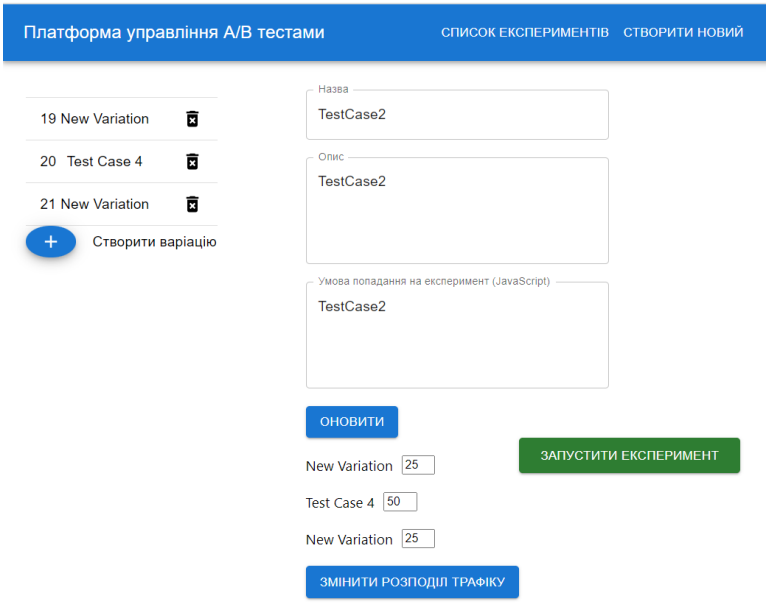



Рисунок В.4 – Результат сценарію №1

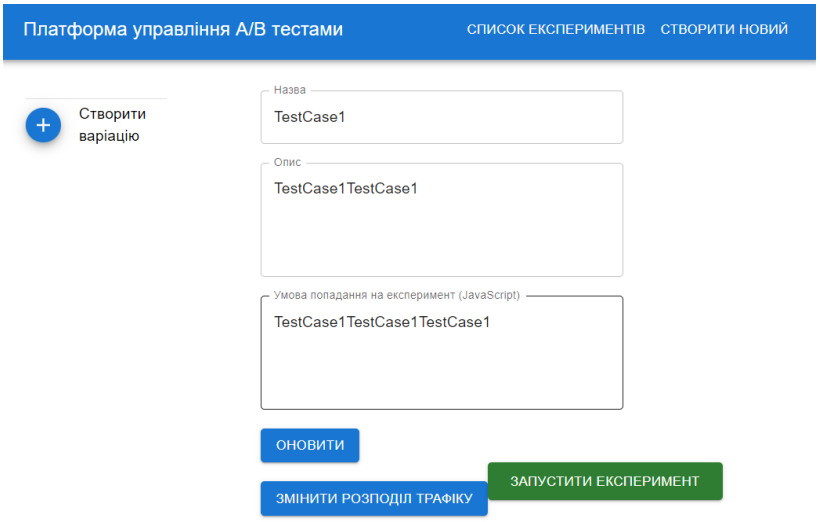



Рисунок В.5 – Результат сценарію №2

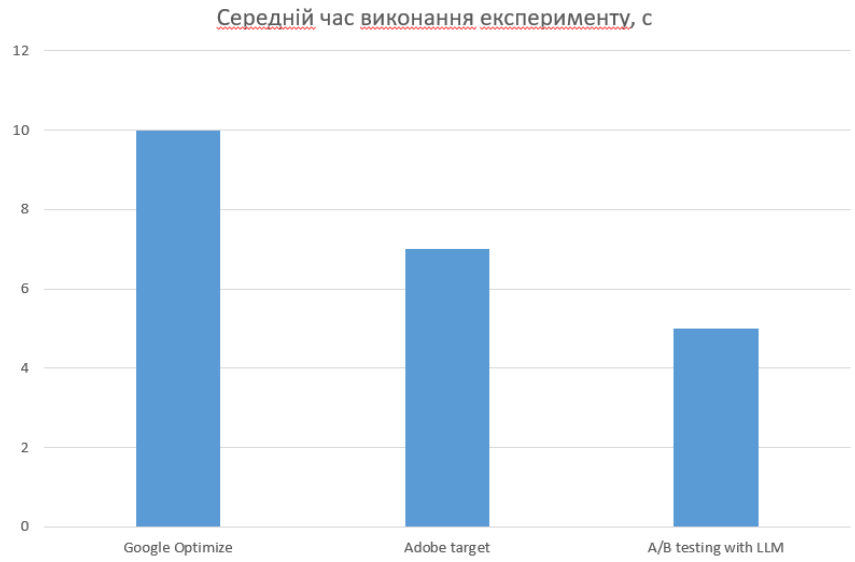



Рисунок В.6 – Результати швидкодії