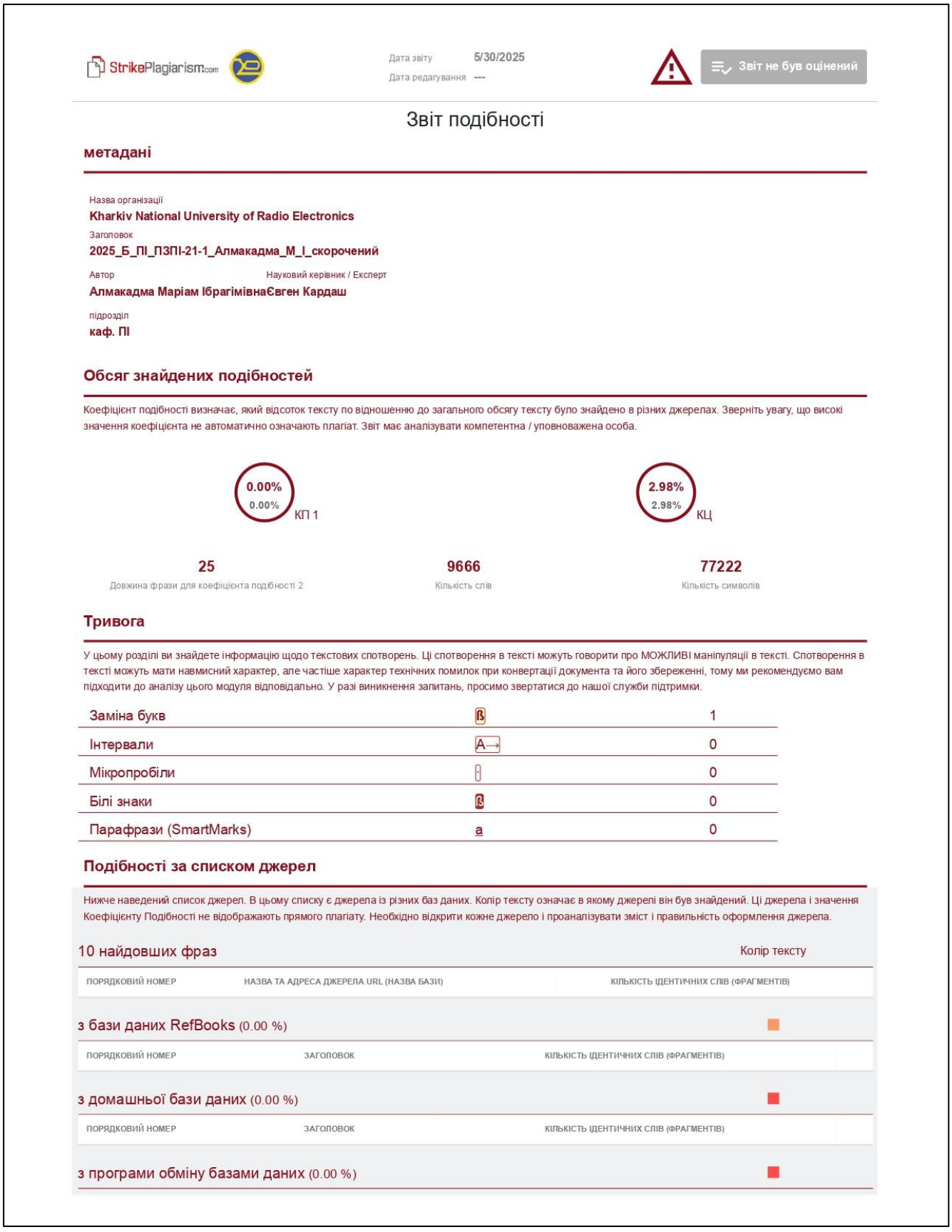
# ДОДАТОК А

## Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



StrikePlagiarism.com

Дата звіту 5/30/2025
Дата редагування ---

Звіт не був оцінений

### Звіт подібності

**метадані**

Назва організації
**Kharkiv National University of Radio Electronics**
Заголовок
**2025_Б_ПІ_ПЗПІ-21-1_Алмакадма_М_І_скорочений**
Автор Науковий керівник / Експерт
**Алмакадма Маріам ІбрагімівнаЄвген Кардаш**
підрозділ
**каф. ПІ**

**Обсяг знайдених подібностей**

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.00% 0.00% КП 1

2.98% 2.98% КЦ

| 25 | 9666 | 77222 |
|---|---|---|
| Довжина фрази для коефіцієнта подібності 2 | Кількість слів | Кількість символів |

**Тривога**

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

| | | |
|---|---|---|
| Заміна букв | ß | 1 |
| Інтервали | A→ | 0 |
| Мікропробіли | · | 0 |
| Білі знаки | ß | 0 |
| Парафрази (SmartMarks) | a | 0 |

**Подібності за списком джерел**

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

| ПОРЯДКОВИЙ НОМЕР | НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ) | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з бази даних RefBooks (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з домашньої бази даних (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з програми обміну базами даних (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) | |
|---|---|---|---|

з Інтернету (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ДЖЕРЕЛО URL | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) | |
|---|---|---|---|

**Список прийнятих фрагментів** (немає прийнятих фрагментів)

| ПОРЯДКОВИЙ НОМЕР | ЗМІСТ | КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ) | |
|---|---|---|---|

ВСТУП

У сучасному світі, де інформаційне навантаження невпинно зростає, дедалі більше людей постають перед труднощами в організації особистого часу, формуванні продуктивних звичок та збереженні мотивації. Відповідно до психологічних досліджень, нестача внутрішньої дисципліни та зовнішньої підтримки часто призводить до прокрастинації, емоційного виснаження й зниження особистої ефективності. У цьому контексті зростає потреба в інноваційних рішеннях, здатних допомогти користувачам ставити цілі, контролювати свій прогрес та досягати бажаних результатів.

Темою кваліфікаційної роботи є створення програмної системи для відстеження своїх досягнень або задач. Система має слугувати персональним цифровим помічником, що сприяє саморозвитку, підвищенню продуктивності та підтримці щоденної мотивації. Основне завдання полягає у впровадженні функціоналу для постановки цілей, фіксації результатів, візуалізації прогресу, а також наданні персоналізованих порад.

Метою роботи є проведення аналізу галузі, формування вимог та планування перевірки якості й надійності цієї програмної системи шляхом комплексного тестування. У межах роботи потрібно протестувати серверну, клієнтську частини, та мобільний застосунок, що забезпечують взаємодію користувача з системою. Тестування охоплює функціональні та нефункціональні аспекти, включаючи навантаження, безпеку, стабільність та зручність використання. Особлива потрібно перевірити цілісність даних, коректність роботи API та адаптивність інтерфейсів. Цільовою аудиторією є користувачі віком від 18 років, які прагнуть покращити особисту ефективність, розвинути нові звички й краще керувати своїм часом.

2

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

У сучасному світі цифрові технології все глибше проникають у повсякденне життя, формуючи основу цифрової економіки. Однією з ключових тенденцій цієї трансформації є зростання зацікавленості до сервісів, що сприяють особистісному розвитку та самостійному плануванню. Зокрема, активно розвивається сегмент застосунків для підвищення особистої продуктивності, які використовують інструменти контролю звичок, мотиваційної підтримки та соціальної взаємодії. Такий напрямок знаходиться на стику психології саморегуляції, гейміфікації та технологічних рішень для здоров'я та навчання.

Доведено, що використання застосунків для формування звичок знижує мотиваційні бар'єри у навчальному процесі, а також підвищує автоматизм поведінки, що робить початок та виконання дій менш залежним від внутрішньої мотивації. У роботі також зазначено, що складні та змінні звички, можуть бути автоматизовані завдяки повторенню у схожому контексті, що забезпечує стабільність і гнучкість в реалізації дій, враховуючи різноманітність змісту. Розглядаючи концепцію мотиваційних конфліктів, бачимо, що суперечливі думки та альтернативні варіанти дій заважають зосередженню на основному завданні. В той час, як автоматизовані звички, які запускаються певним контекстом або сигналом, допомагають зменшити частоту таких конфліктів та забезпечують безперервність дій та ментальне розвантаження [1].

Швидкий розвиток цифрових технологій в освіті спричинив трансформацію традиційних підходів до викладання, сприяючи переходу до активного, орієнтованого на студента навчання. У контексті концепції Education 4.0 ключовим є впровадження технологій, які відповідають потребам сучасного суспільства та

ДОДАТОК Б

Слайди презентації

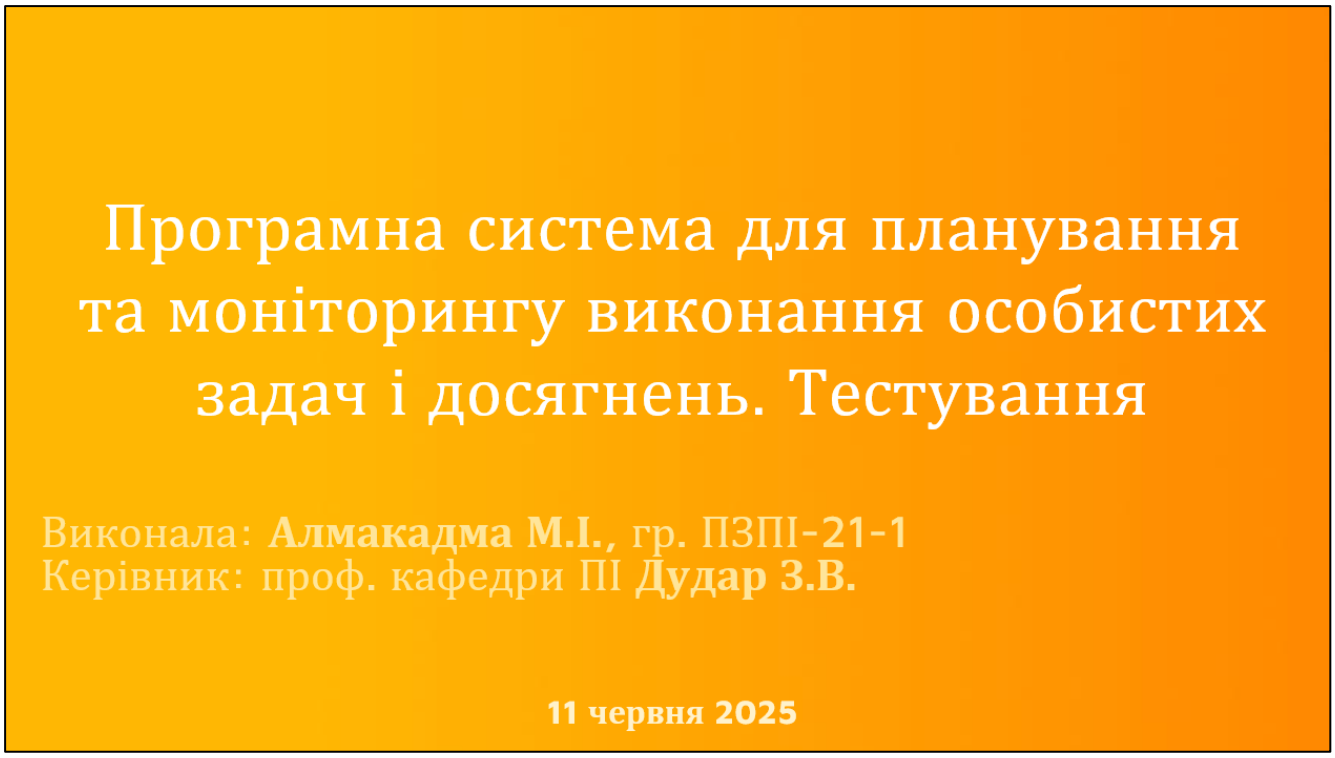



Рисунок Б.1 – Слайд 1

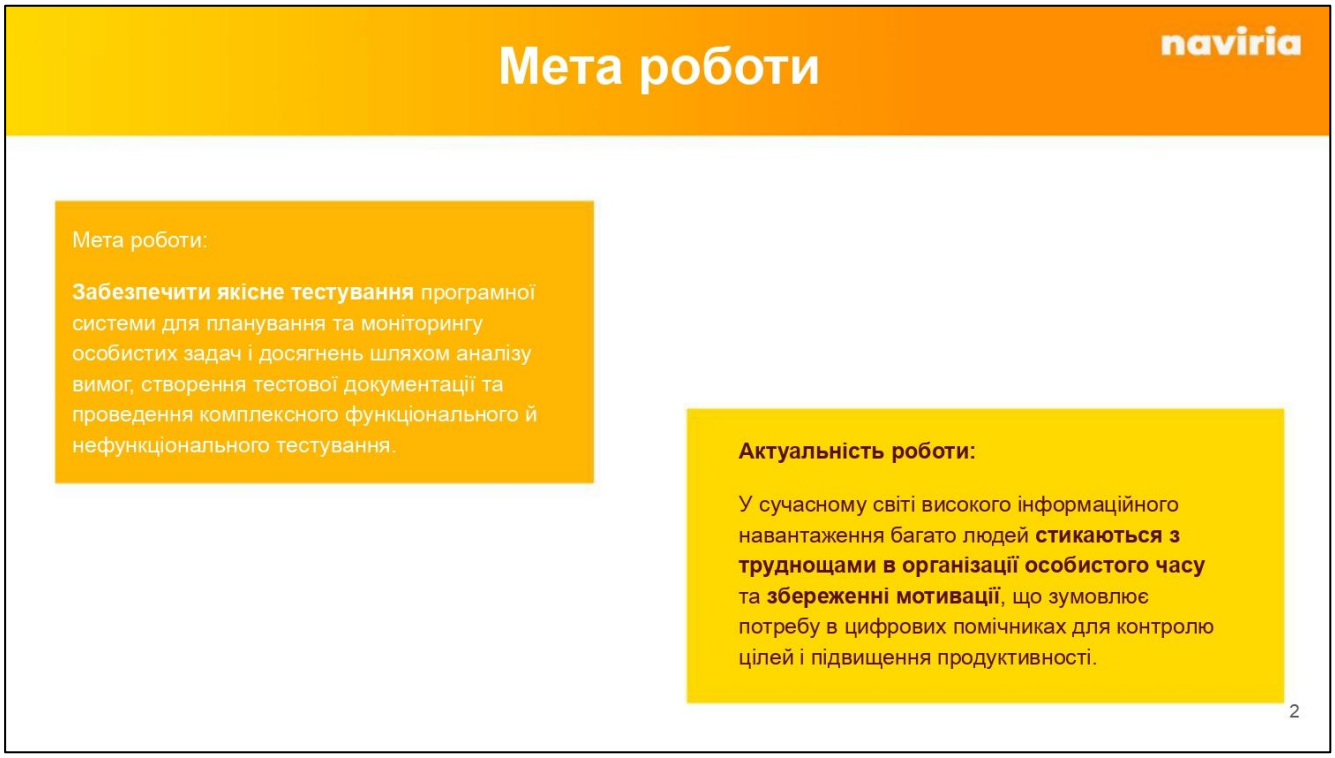



Рисунок Б.2 – Слайд 2

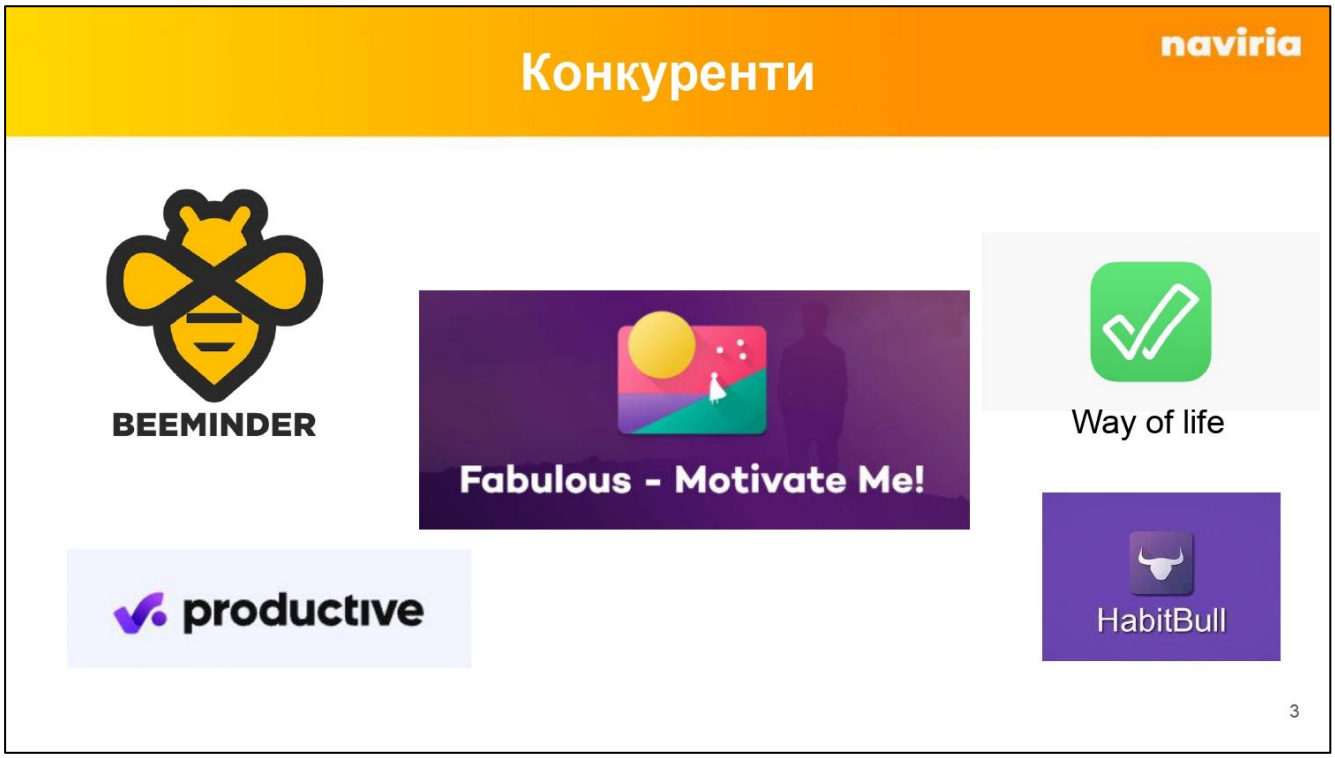



Рисунок Б.3 – Слайд 3


| Критерії | Beeminder | HabitBull | Way of Life | Productive | Fabulous |
|---|---|---|---|---|---|
| Платформи | Web, iOS, Android, Slack | iOS, Android | iOS, Android | iOS, Android | iOS, Android |
| Вартість | Безкоштовно/преміум $8/міс | Безкоштовно, преміум доступ $4.99/міс. | Безкоштовно/$6.90 одноразово | Безкоштовно, преміум доступ $9/міс. | Безкоштовно, преміум доступ $39.99/рік. |
| Гейміфікація | Так (штрафи, система мотивації) | Так (гейміфікація звичок) | - | Так (балли, досягнення) | Так (рутини, нагороди) |
| Соціалізація | Низька (обмежена взаємодія) | Низька | Низька | Низька | - |
| AI | - | - | - | - | Так |
| Інтерфейс | Простий, мінімалістичний, орієнтований на графіки та дані | Простий, з графіками і трекінгом звичок | Простий, з графіками і трекінгом звичок | Мінімалістичний, з фокусом на звички | Естетично привабливий, автентичний дизайн |
| Нагадування | Так (регулярні нагадування про прогрес і порушення цілей) | Так (індивідуальні нагадування) | Так (підтримка регулярних нагадувань для звичок) | Так (підтримка нагадувань для звичок) | Так (регулярні нагадування) |

Рисунок Б.4 – Слайд 4

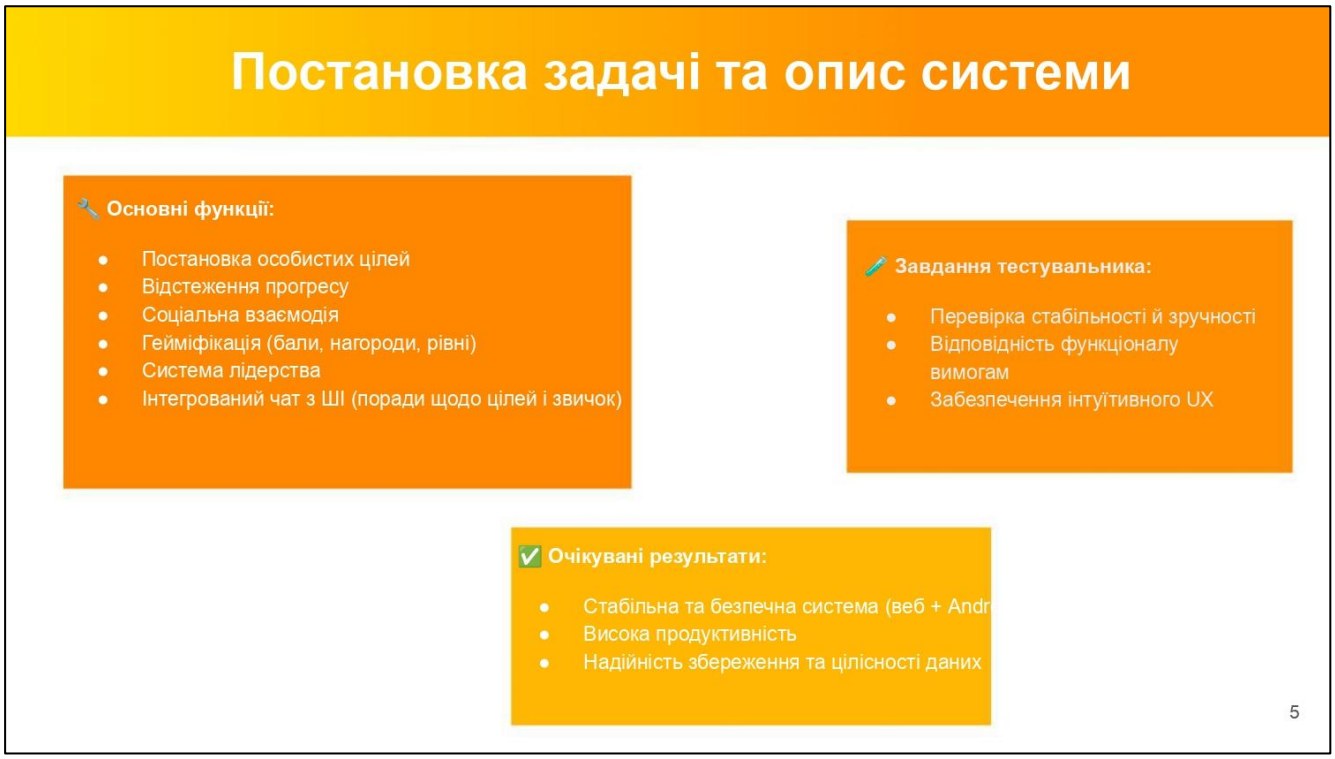



Рисунок Б.5 – Слайд 5

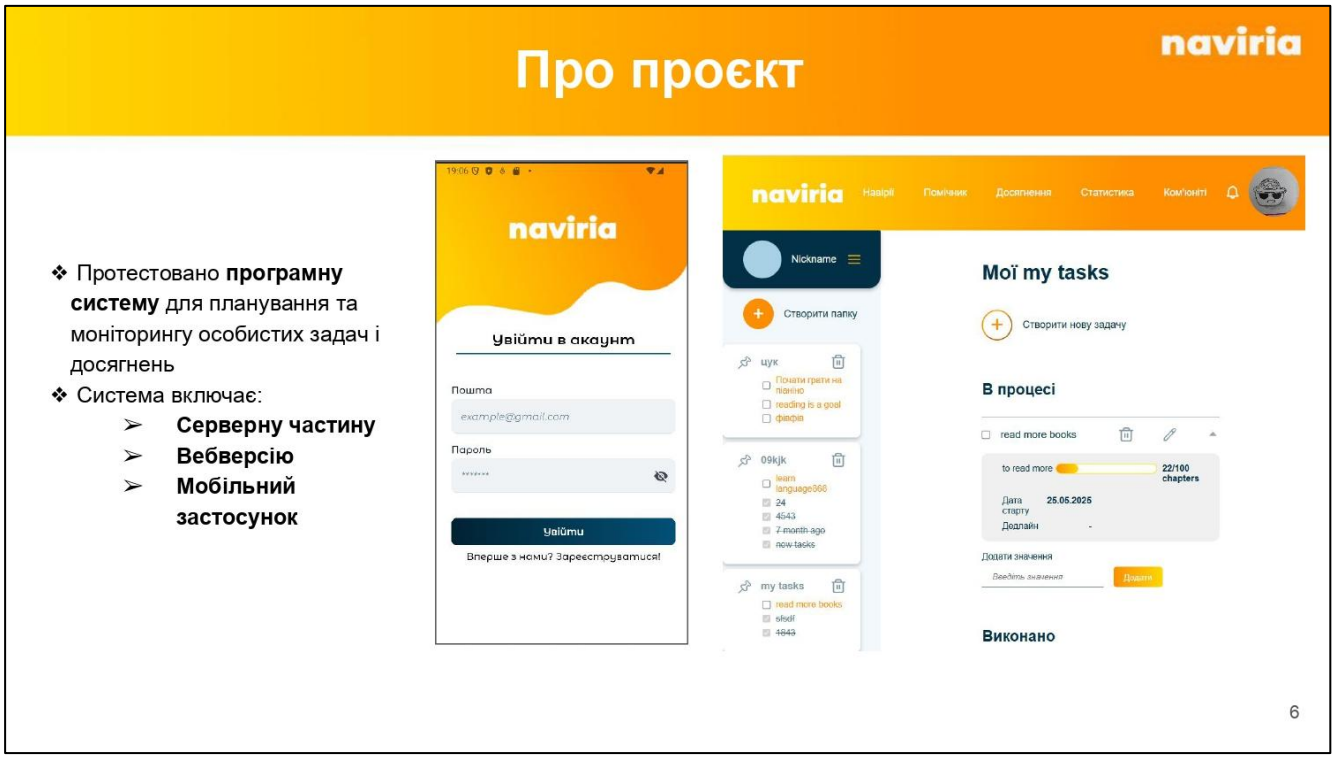
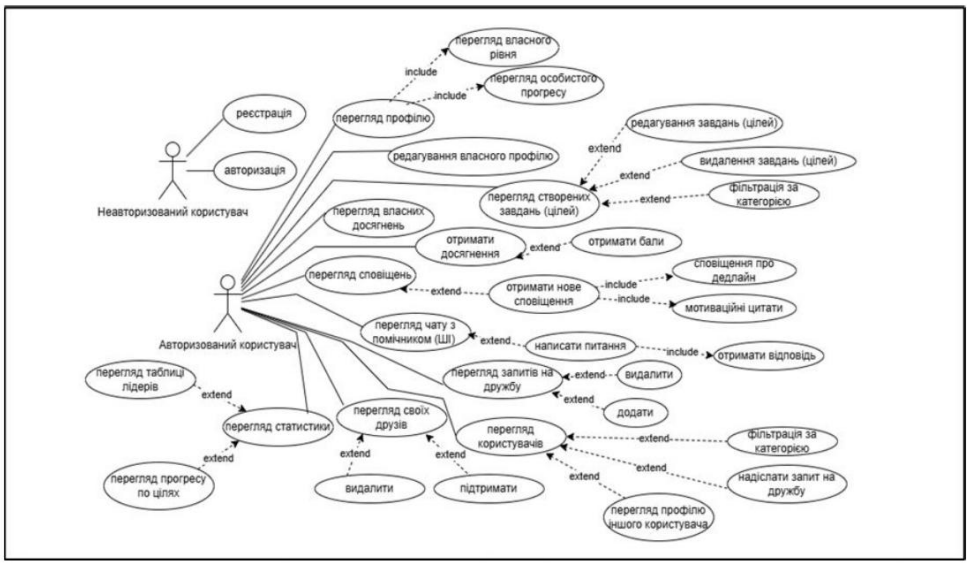



Рисунок Б.6 – Слайд 6

naviria

# Діаграма прецедентів з актором користувач

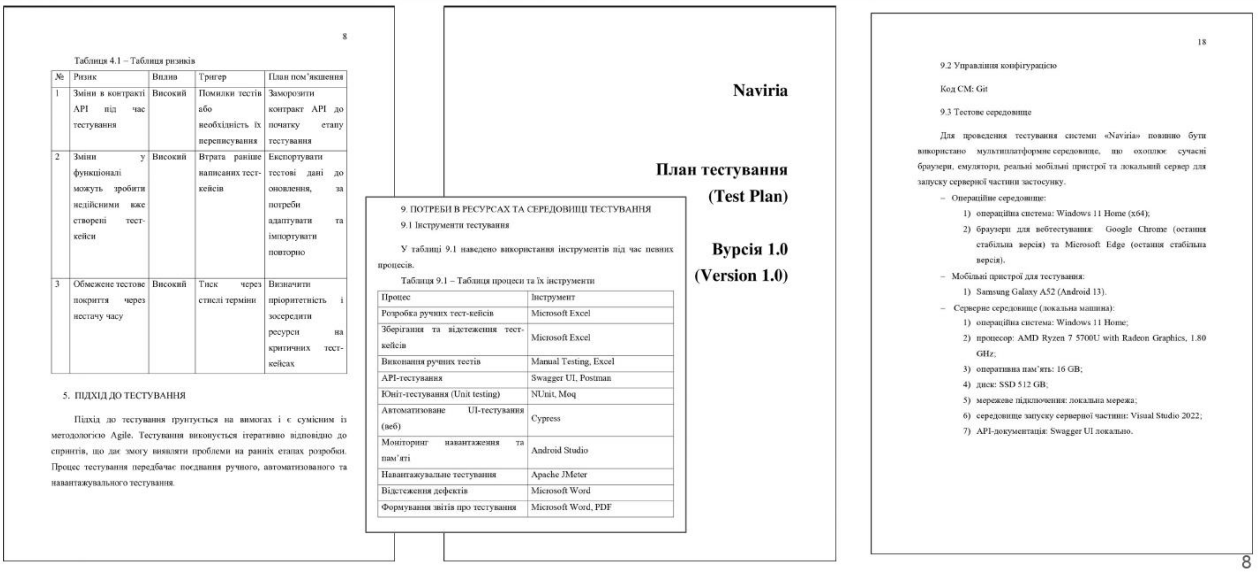



7

Рисунок Б.7 – Слайд 7

naviria

# Тест-план



8

Таблиця 4.1 – Таблиця ризиків

| № | Ризик | Вплив | Тригер | План пом'якшення |
|---|---|---|---|---|
| 1 | Зміни в контракті API під час тестування | Високий | Помилки тестів або необхідність їх переписування | Заморозити контракт API до початку етапу тестування |
| 2 | Зміни у функціоналі можуть зробити недійсними вже створені тест-кейси | Високий | Втрата раніше написаних тест-кейсів | Експортувати тестові дані до оновлення, за потреби адаптувати та імпортувати повторно |
| 3 | Обмеження тестове покриття через нестачу часу | Високий | Тиск через стислі терміни | Визначити пріоритетність і зосередити ресурси на критичних тест-кейсах |

5. ПІДХІД ДО ТЕСТУВАННЯ

Підхід до тестування ґрунтується на вимогах і є сумісним із методологією Agile. Тестування виконується ітеративно відповідно до спринтів, що дає змогу виявляти проблеми на ранніх етапах розробки. Процес тестування передбачає поєднання ручного, автоматизованого та навантажувального тестування.

**Naviria**

**План тестування**
**(Test Plan)**

**Версія 1.0**
**(Version 1.0)**

9. ПОТРЕБИ В РЕСУРСАХ ТА СЕРЕДОВИЩІ ТЕСТУВАННЯ

9.1 Інструменти тестування

У таблиці 9.1 наведено використання інструментів під час певних процесів.

Таблиця 9.1 – Таблиця процеси та їх інструменти

| Процес | Інструмент |
|---|---|
| Розробка ручних тест-кейсів | Microsoft Excel |
| Зберігання та відстеження тест-кейсів | Microsoft Excel |
| Виконання ручних тестів | Manual Testing, Excel |
| API-тестування | Swagger UI, Postman |
| Юніт-тестування (Unit testing) | NUnit, Moq |
| Автоматизоване UI-тестування (веб) | Cypress |
| Моніторинг навантаження та пам'яті | Android Studio |
| Навантажувальне тестування | Apache JMeter |
| Відстеження дефектів | Microsoft Word |
| Формування звітів про тестування | Microsoft Word, PDF |

18

9.2 Управління конфігурацією

Код CM: Git

9.3 Тестове середовище

Для проведення тестування системи «Naviria» повинно бути використано мультиплатформне середовище, що охоплює сучасні браузери, емулятори, реальні мобільні пристрої та локальний сервер для запуску серверної частини застосунку.

- Операційне середовище:
  1) операційна система: Windows 11 Home (x64);
  2) браузери для вебтестування: Google Chrome (остання стабільна версія) та Microsoft Edge (остання стабільна версія).
- Мобільні пристрої для тестування:
  1) Samsung Galaxy A52 (Android 13).
- Серверне середовище (локальна машина):
  1) операційна система: Windows 11 Home;
  2) процесор: AMD Ryzen 7 5700U with Radeon Graphics, 1.80 GHz;
  3) оперативна пам'ять: 16 GB;
  4) диск: SSD 512 GB;
  5) мережеве підключення: локальна мережа;
  6) середовище запуску серверної частини: Visual Studio 2022;
  7) API-документація: Swagger UI локально.

8

Рисунок Б.8 – Слайд 8

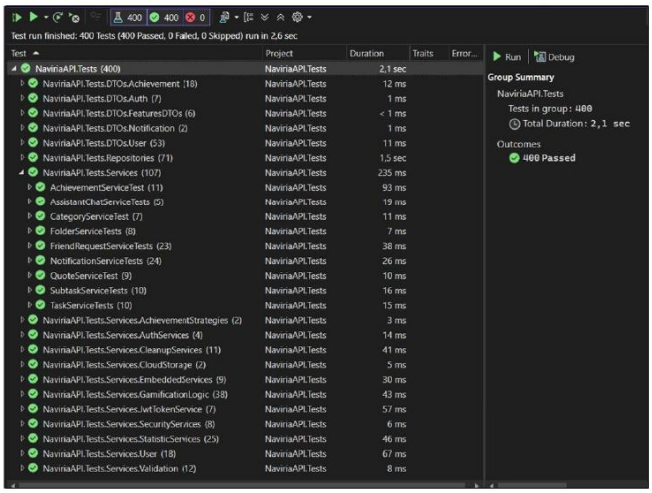
## Опис прийнятих програмних рішень та їх тестування
## Серверна частина (Back-end)

Автоматизоване тестування (C#, **NUnit**):

• Unit-тести – перевірка бізнес-логіки (services)

• Інтеграційні тести – перевірка взаємодії з MongoDB

• Регресійне тестування – повторне використання тих самих тестів після оновлень




9

Рисунок Б.9 – Слайд 9

## Ручне тестування API (Swagger UI)

naviria

🔍 Що тестувалось:

• Всі основні CRUD-запити (GET, POST, PUT, DELETE)

• Контролери: досягнення, чат з помічником, авторизація, категорії, папки, друзі, таблиця лідерів, сповіщення, цитати, задачі, підзадачі, пошук тощо

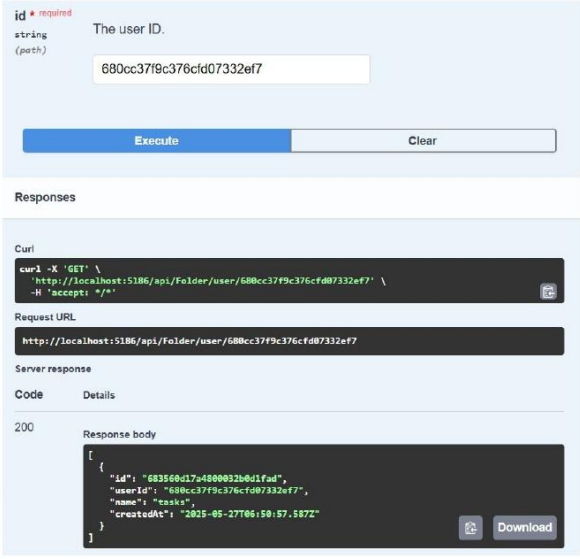
🛠️ Як тестувалось:

• Виконання запитів через Swagger UI

• Перевірка змін у MongoDB після кожного запиту

• Контроль статусів HTTP, структури відповіді та бізнес-логіки

mongoDB




10

Рисунок Б.10 – Слайд 10

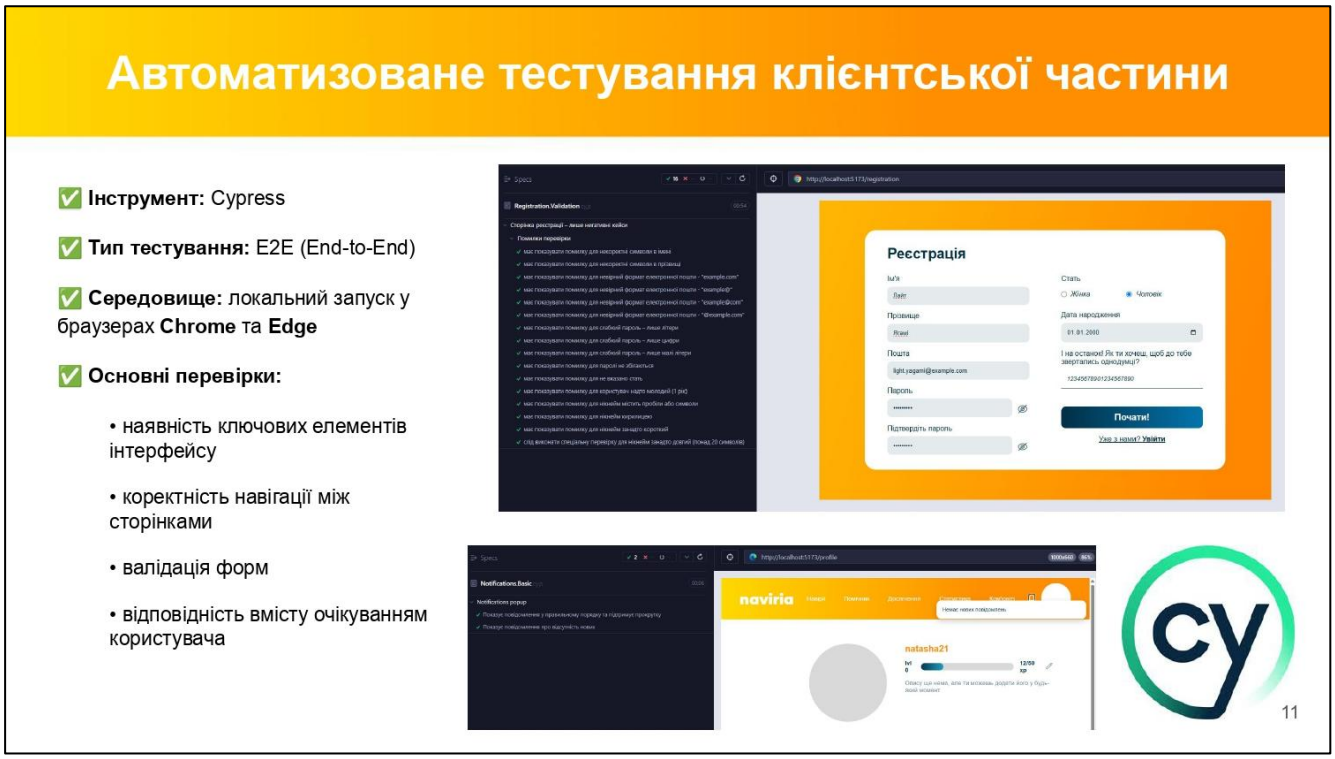



Рисунок Б.11 – Слайд 11

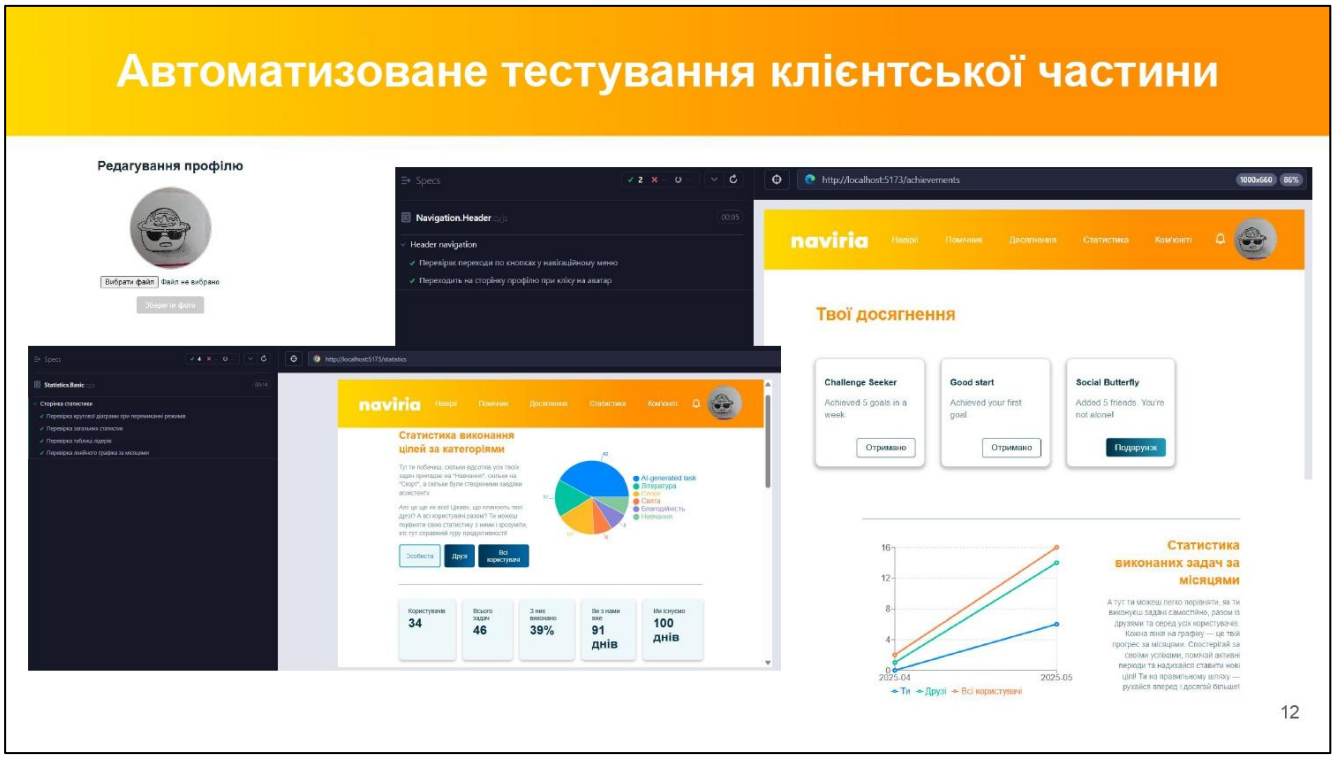
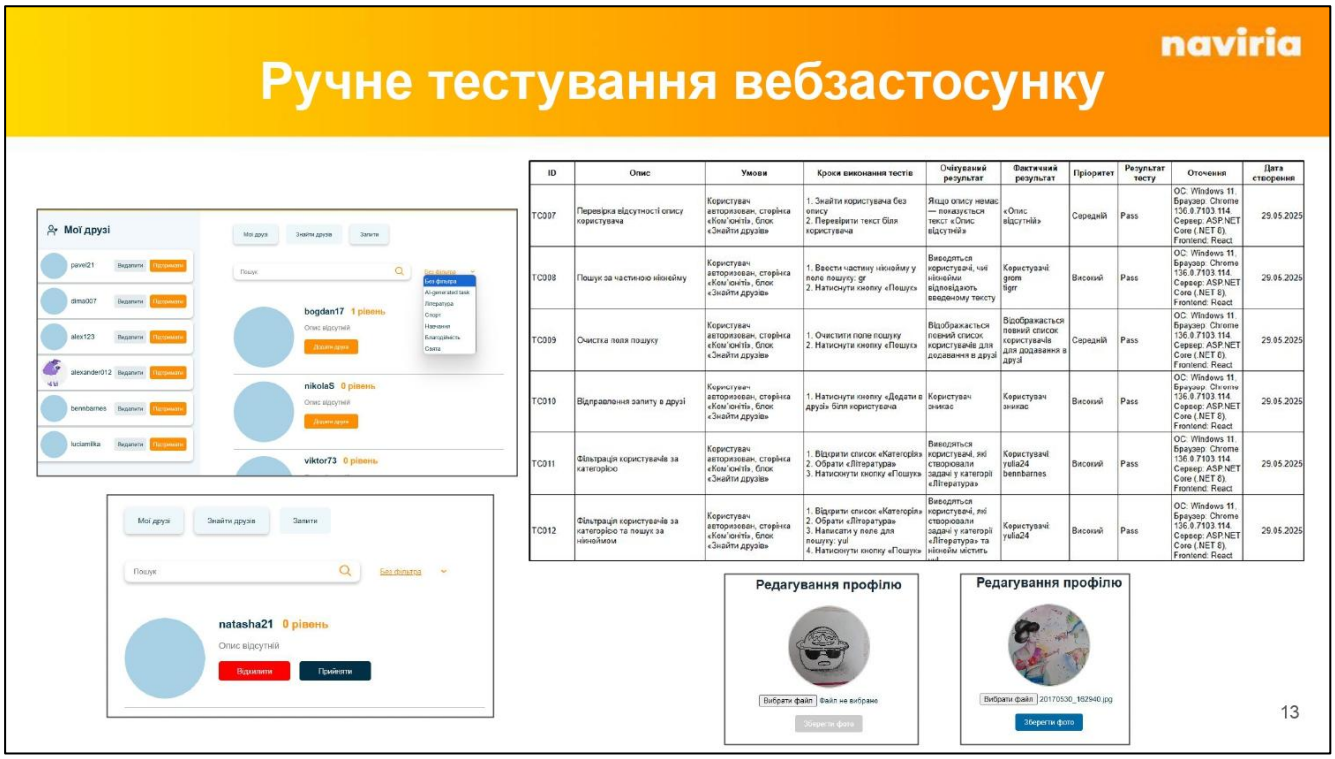
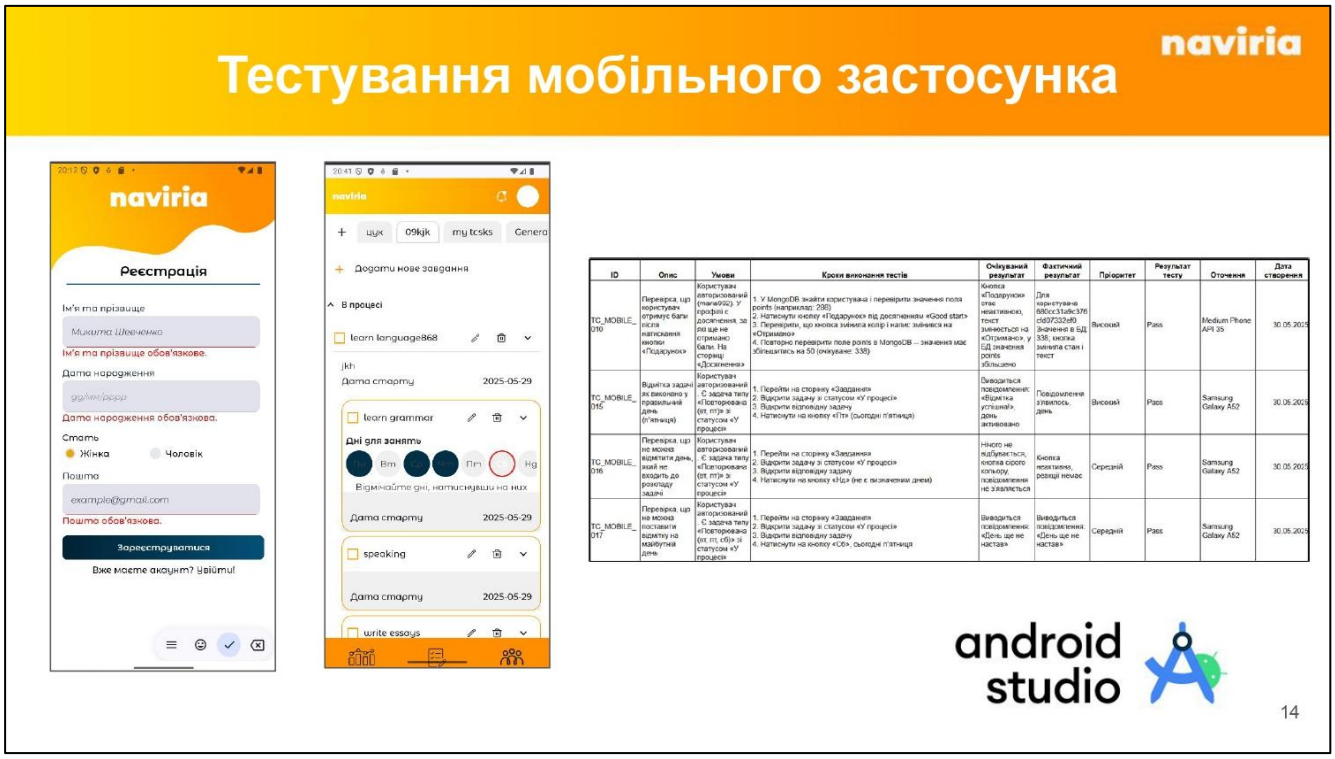



Рисунок Б.12 – Слайд 12




| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC007 | Перевірка відсутності опису користувача | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Знайти користувача без опису<br>2. Перевірити текст біля користувача | Якщо опису немає — показується текст «Опис відсутній» | «Опис відсутній» | Середній | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |
| TC008 | Пошук за частиною нікнейму | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Ввести частину нікнейму у поле пошуку: gr<br>2. Натиснути кнопку «Пошук» | Виводяться користувачі, чиї нікнейми відповідають введеному тексту | Користувачі: grom tigrr | Високий | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |
| TC009 | Очистка поля пошуку | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Очистити поле пошуку<br>2. Натиснути кнопку «Пошук» | Відображається повний список користувачів для додавання в друзі | Відображається повний список користувачів для додавання в друзі | Середній | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |
| TC010 | Відправлення запиту в друзі | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Натиснути кнопку «Додати в друзі» біля користувача | Користувач зникає | Користувач зникає | Високий | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |
| TC011 | Фільтрація користувачів за категорією | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Відкрити список «Категорія»<br>2. Обрати «Література»<br>3. Натиснути кнопку «Пошук» | Виводяться користувачі, які створювали задачі у категорії «Література» | Користувачі: yulia24 bennbarnes | Високий | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |
| TC012 | Фільтрація користувачів за категорією та пошук за нікнеймом | Користувач авторизований, сторінка «Комʼюніті», блок «Знайти друзів» | 1. Відкрити список «Категорія»<br>2. Обрати «Література»<br>3. Написати у поле для пошуку: yul<br>4. Натиснути кнопку «Пошук» | Виводяться користувачі, які створювали задачі у категорії «Література» та нікнейм містить yul | Користувачі: yulia24 | Високий | Pass | ОС: Windows 11<br>Браузер: Chrome 136.0.7103.114<br>Сервер: ASP.NET Core (.NET 8)<br>Frontend: React | 29.05.2025 |

Рисунок Б.13 – Слайд 13

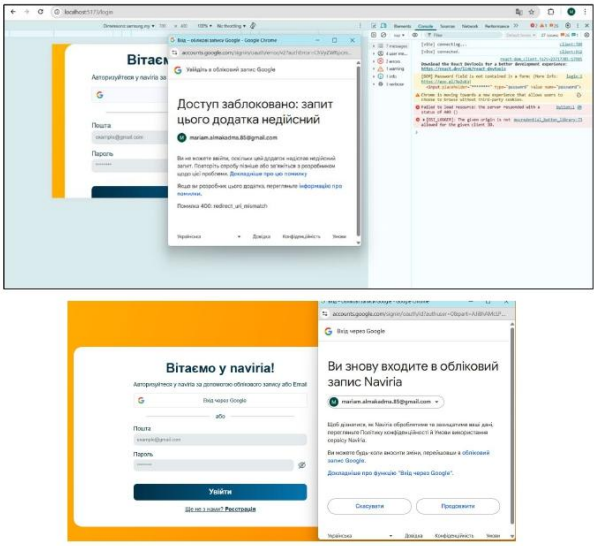



| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC_MOBILE_010 | Перевірка, що користувач отримує бали після натискання кнопки «Подарунок» | Користувач авторизований (email@gc). У профілі є досягнення, за яке ще не отримано бали. На сторінці «Досягнення» | 1. У MongoDB знайти користувача і перевірити значення поля points (наприклад: 298)<br>2. Натиснути кнопку «Подарунок» від досягнення «Good start»<br>3. Перевірити, що кнопка зникла колір і напис змінився на «Отримано»<br>4. Повторно перевірити поле points в MongoDB — значення має збільшитись на 50 (очікуване: 338) | Кнопка «Подарунок» стає неактивною, текст змінюється на «Отримано», у БД значення points збільшено | Для користувача 68bcc31a8c376cb97232e80 значення в БД 338, кнопка змінила стан і текст | Високий | Pass | Medium Phone API 35 | 30.05.2025 |
| TC_MOBILE_015 | Відмітка задачі як виконаної у правильний день (п'ятниця) | Користувач авторизований. Є задача типу «Повторювана» (пт, пт)к зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Пт» (сьогодні п'ятниця) | Виводиться повідомлення: «Відмітка успішна!», день активовано | Повідомлення з'явилось, день | Високий | Pass | Samsung Galaxy A52 | 30.05.2025 |
| TC_MOBILE_016 | Перевірка, що не можна відмітити день, який не входить до розкладу задачі | Користувач авторизований. Є задача типу «Повторювана» (пт, пт)к зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Нд» (не є визначеним днем) | Нічого не відбувається, кнопка сірого кольору, повідомлення не з'являється | Кнопка неактивна, реакції немає | Середній | Pass | Samsung Galaxy A52 | 30.05.2025 |
| TC_MOBILE_017 | Перевірка, що не можна поставити відмітку на майбутній день | Користувач авторизований. Є задача типу «Повторювана» (пт, пт, сб)о зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Сб», сьогодні п'ятниця | Виводиться повідомлення: «День ще не настав» | Виводиться повідомлення: «День ще не настав» | Середній | Pass | Samsung Galaxy A52 | 30.05.2025 |

Рисунок Б.14 – Слайд 14

naviria

# Життєвий цикл дефекту




| Поле | Опис |
|---|---|
| Ідентифікатор дефекту | BUG-004 |
| Короткий опис | Помилка авторизації через Google: redirect_url_mismatch |
| Кроки для відтворення | 1. Запустити сервер (http)<br>2. Запустити вебсайт<br>3. Відкрити сторінку входу: localhost:5173/login<br>4. Натиснути кнопку «Вхід через Google» |
| Очікуваний результат | Користувач успішно авторизується через Google-акаунт. |
| Фактичний результат | Помилка 400: redirect_url_mismatch. |
| Пріоритет | 1 – Обов'язково виправити |
| Компонент | Авторизація / Сервер |
| Мітки (Labels) | auth, google_login, critical |
| Коментарі | Користувач вже зареєстрований з цією поштою. Помилка виникає на боці серверу при неправильній конфігурації OAuth. |
| Докази (Screenshots) | Дивитись рисунок 4.13 |
| Виконавець (Assignee) | Призначено розробнику сервера (Черненко А.) |
| Серйозність | 2 – Висока |
| Логи | У консолі панелі розробника: the given origin is not allowed for the given client ID |
| Час тестування | 10:00 07.05.2025 |
| Тестове середовище | ОС: Windows 11, Браузер: Chrome 136.0.7103.114, Сервер: ASP.NET Core (.NET 8), Frontend: React |
| Тестові дані | Google акаунт: mariam.almakadma.85@gmail.com |

15

Рисунок Б.15 – Слайд 15

naviria

# Продуктивність системи

**Сервер**

- Стабільна пропускна здатність: **>80 запитів/сек** до 500 користувачів
- Оброблено **10 000 запитів** при 1000 користувачах
- Зростання часу відповіді вказує на **межу поточної конфігурації**, але система залишається працездатною

**Вебзастосунок**

- **Apdex ~1.0** та **0% помилок** до 400 користувачів
- Середній час відповіді: **<314 мс**
- Максимальна пропускна здатність: **>570 запитів/сек** при 1000 користувачах
- При перевантаженні (1000 користувачів): **5.4% помилок**, **Apdex — 0.27**

**Мобільний застосунок**

- CPU: **0.01%**, ~43 потоки
- Пам'ять: **162,7 МБ**, домінують **рендеринг та JIT**
- **Можливі витоки**, але **збирання стабільне**

APACHE JMeter™

16

Рисунок Б.16 – Слайд 16


Рисунок Б.17 – Слайд 17

ДОДАТОК В

Тест план

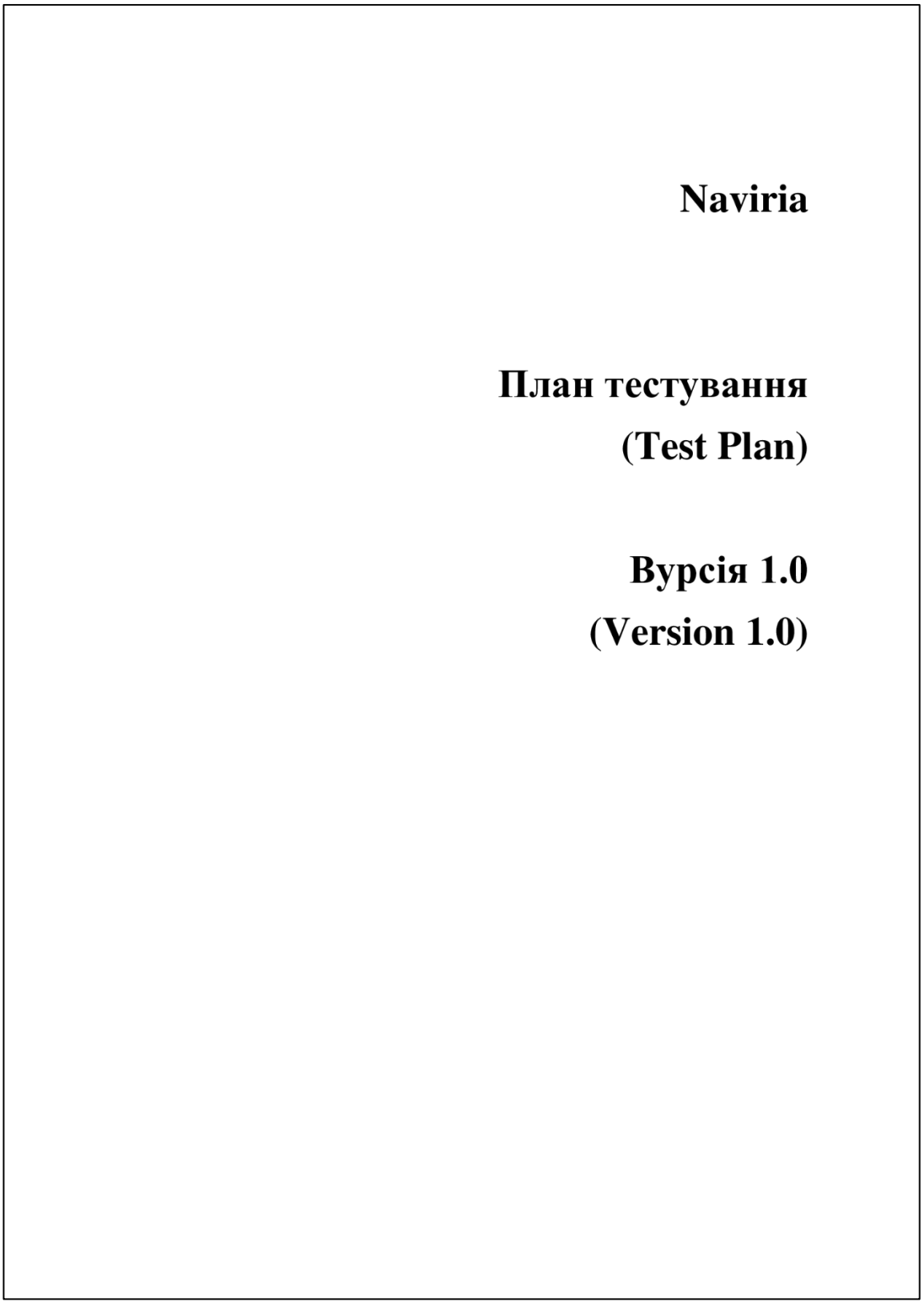



Рисунок В.1 – Тест-план сторінка 1

Історія змін документа

| Дата | Версія | Опис | Автор |
|---|---|---|---|
| 2025-05-10 | 1.0 | Початкове створення тестового плану | Маріам Алмакадма |

Рисунок В.2 – Тест-план сторінка 2

ЗМІСТ



Рисунок В.3 – Тест-план сторінка 3

Рисунок В.4 – Тест-план сторінка 4

# 1. ВСТУП

Цей Тестовий План описує стратегію забезпечення якості для проєкту Naviria, включаючи серверний API (NaviriaAPI на C#), вебінтерфейс (React) і мобільний застосунок для Android. Містить цілі тестування, підходи, ролі, середовища та графіки, щоб забезпечити стабільну роботу всіх компонентів системи відповідно до вимог користувачів.

## 1.1 Члени команди

У таблиці 1.1 наведено список членів команди та їх ролі.

Таблиця 1.1 – Таблиця членів команди

| Ім’я та Прізвище | Посада |
|---|---|
| Маріам Алмакадма | QA Engineer |
| Анастасія Червенко | C# Developer |
| Надія Дашко | Frontend Developer |
| Єлизавета Попова | Mobile Developer (Android) |

# 2. ОБСЯГ ТЕСТУВАННЯ

## 2.1 Функціонал, що підлягає тестуванню

До функціоналу, який підлягає тестуванню в межах проєкту «Naviria», належать як функціональні, так і нефункціональні вимоги. Зокрема, буде перевірено наступне:

Функціональні вимоги:

- реєстрація та вхід користувачів (auth flow);
- управління профілем (досягнення, статистика);
- управління цілями (завдання);
- аналітика та відстеження прогресу (графіки);
- гейміфікація (бали, рівні, таблиця лідерів);

Рисунок В.5 – Тест-план сторінка 5

– сповіщення;
– чат зі ШІ;
– соціальні функції (друзі, виклики);
– робота з базою даних (інтеграція з MongoDB);
– API-функціональність (валідація відповідей);
– контроль доступу (перевірка автентифікації);
– обробка помилок (400, 404, 500).

Нефункціональні вимоги:

– продуктивність;
– масштабованість;
– стабільність роботи та відсутність критичних збоїв;
– зручність користування інтерфейсом у браузері та на мобільному пристрої;
– правильна реалізація авторизації, захист персональних даних;
– поведінка під навантаженням.

2.2 Функціонал, що не тестується

– все, що не зазначено у пункті 2.1.

3. ЦІЛІ ЯКОСТІ

3.1 Основні цілі

Основною метою забезпечення якості в межах проєкту «Naviria» є гарантування того, що всі функціональні та нефункціональні вимоги до системи виконуються повною мірою.

Особлива увага приділяється стабільності та масштабованості серверної частини і вебінтерфейсу при роботі під навантаженням, що дозволяє забезпечити надійну роботу системи в умовах високої інтенсивності користувацьких запитів.

Рисунок В.6 – Тест-план сторінка 6

Не менш важливим є забезпечення безперебійної інтеграції між серверною частиною, вебінтерфейсом та мобільним застосунком, що дозволяє досягти узгодженості функціонування всіх компонентів. Крім того, кінцевою метою є підтвердження готовності програмного забезпечення до випуску шляхом досягнення відповідності всім вимогам і очікуванням зацікавлених сторін.

3.2 Додаткові цілі

Серед другорядних, але важливих завдань є раннє виявлення дефектів на етапах розробки, що дозволяє мінімізувати витрати на їх усунення. Ефективна комунікація в межах команди, зокрема оперативна передача інформації про дефекти, сприяє підвищенню швидкості реакції на виявлені проблеми. Також важливим є забезпечення широкого покриття тестами як основного функціоналу, так і граничних сценаріїв використання, що дозволяє досягти більшої впевненості в надійності програмного продукту.

4. ПРИПУЩЕННЯ ТА РИЗИКИ

4.1 Припущення

- код буде структурованим та викладеним у GitHub;
- основне тестування проводиться з використанням реальної бази даних, за винятком деяких випадків, де застосовуються ізольовані тестові дані для запобігання впливу на продуктивне середовище;
- API задокументовано у Swagger.

4.2 Ризики

Було визначено ризики та відповідні дії для пом’якшення їхнього впливу на проєкт. Вплив ризику визначається тим, як проєкт постраждає у разі його спрацювання. Тригером є те, яка віха або подія призведе до того, що ризик стане проблемою, яку потрібно вирішити (див. табл. 4.1).

Рисунок В.7 – Тест-план сторінка 7

Таблиця 4.1 – Таблиця ризиків

| № | Ризик | Вплив | Тригер | План пом’якшення |
|---|---|---|---|---|
| 1 | Зміни в контракті API під час тестування | Високий | Помилки тестів або необхідність їх переписування | Заморозити контракт API до початку етапу тестування |
| 2 | Зміни у функціоналі можуть зробити недійсними вже створені тест-кейси | Високий | Втрата раніше написаних тест-кейсів | Експортувати тестові дані до оновлення, за потреби адаптувати та імпортувати повторно |
| 3 | Обмежене тестове покриття через нестачу часу | Високий | Тиск через стислі терміни | Визначити пріоритетність і зосередити ресурси на критичних тест-кейсах |

## 5. ПІДХІД ДО ТЕСТУВАННЯ

Підхід до тестування ґрунтується на вимогах і є сумісним із методологією Agile. Тестування виконується ітеративно відповідно до спринтів, що дає змогу виявляти проблеми на ранніх етапах розробки. Процес тестування передбачає поєднання ручного, автоматизованого та навантажувального тестування.

Рисунок В.8 – Тест-план сторінка 8

5.1 Ручне тестування

- тестування API через Swagger UI: виконується для більшості кінцевих точок з метою перевірки правильності відповіді, формату та обробки крайових випадків;
- поверхневе тестування вебінтерфейсу та мобільних застосунків: зосереджене на компонентах, які зазнали змін або є новими;
- валідація UI/UX взаємодії: базове тестування інтерфейсу з метою перевірки візуальної узгодженості та навігації у змінених частинах системи.

5.2 Автоматизоване тестування

- логіка серверної частини та валідація: автоматизується за допомогою фреймворків NUnit і Moq для перевірки бізнес-логіки та серверної обробки даних;
- інтеграційне тестування з MongoDB: охоплює сценарії взаємодії з базою даних для перевірки узгодженості та правильності збереження даних;
- повторювані UI-тести для вебплатформ: автоматизовані тести з використанням Cypress (для веб) для перевірки валідності форм, навігаційних переходів і статичних очікуваних результатів.

6. КРИТЕРІЇ ПОЧАТКУ ТА ЗАВЕРШЕННЯ ТЕСТУВАННЯ

6.1 Критерії початку тестування

Тестування може бути розпочато лише за умови дотримання таких передумов:

- вихідний код проєкту розміщено у репозиторії GitHub;

Рисунок В.9 – Тест-план сторінка 9

– усі необхідні програмні засоби, включаючи інструменти тестування, успішно встановлено та перевірено на працездатність;
– тестове середовище налаштовано й готове до використання (серверна частина, база даних, мобільний емулятор);
– надано повну документацію до вимог, а також специфікацію API у форматі Swagger;
– підготовлено тестові дані відповідно до очікуваних сценаріїв;
– тестувальник повністю ознайомлена з вимогами та логікою функціонування системи.

6.2 Критерії завершення тестування

Тестування вважається завершеним, якщо виконано такі умови:
– усі критичні тест-кейси були виконані;
– не залишилося жодних відкритих дефектів із високим або блокуючим рівнем серйозності;
– досягнуто охоплення вимог тестами на рівні не менше ніж 95%;
– підготовлено та переглянуто фінальний звіт про результати тестування.

7. КРИТЕРІЇ ПРИЗУПИНЕННЯ ТА ВІДНОВЛЕННЯ ТЕСТУВАННЯ

7.1 Критерії призупинення тестування

До критеріїв призупинення тестування відноситься:
– невдала збірка застосунку або наявність критичних дефектів, що блокують подальше тестування;
– відсутність доступу до тестового середовища, що унеможливлює виконання запланованих перевірок;

Рисунок В.10 – Тест-план сторінка 10

– суттєві зміни у вимогах під час активного спринту, що призводять до необхідності перегляду наявних тест-кейсів або тестової стратегії.

7.2 Критерії відновлення тестування

До критеріїв відновлення тестування відноситься повне усунення блокуючих дефектів та стабілізації тестового середовища.

8. СТРАТЕГІЯ ТЕСТУВАННЯ

Процес тестування в межах проєкту «Navigia» реалізується одним спеціалістом із забезпечення якості, який виконує повний цикл тестування – від аналізу вимог до складання підсумкової звітності.

Стратегія передбачає комплексну діяльність, спрямовану на забезпечення якості розроблюваного програмного забезпечення.

8.1 Роль тестувальника у процесі тестування

Тестувальник повинен виконати наступні дії:

– аналіз вимог: тестувальник ознайомлюється з функціональними вимогами, що надаються розробниками, зокрема у вигляді специфікацій або Swagger-документації. На основі цього виконується повне розуміння очікуваного функціоналу;
– підготовка тест-кейсів: тест-кейси формуються на основі аналізу вимог, а також результатів попереднього дослідницького тестування. Усі можливі сценарії, що охоплюють як позитивні, так і негативні випадки, документуються у структурованому вигляді;
– створення тестових даних: тестувальник самостійно генерує тестові дані відповідно до обраних сценаріїв та умов тестування, використовуючи розроблене середовище (локальне або тестове);

Рисунок В.11 – Тест-план сторінка 11

- виконання тестів: тестування проводиться вручну та за допомогою автоматизованих засобів (NUnit, Cypress, тощо) відповідно до тест-кейсів. Після кожного запуску фіксуються фактичні результати виконання (успішно/неуспішно) у відповідній документації;
- реєстрація дефектів: усі виявлені дефекти документуються у системі відстеження із зазначенням умов відтворення, очікуваного та фактичного результату. Про виявлені дефекти повідомляється відповідальному розробнику;
- повторне тестування і регресія: після усунення дефектів розробником виконується повторне тестування (retesting). У разі, якщо зміни можуть вплинути на інші частини системи, додатково проводиться регресійне тестування для перевірки суміжного функціоналу;
- звітність та передача результатів: після завершення всіх етапів тестування формується підсумковий звіт про якість із деталізацією тест-кейсів, покриття вимог, кількості виявлених і виправлених дефектів. Звіт зберігається локально та, за потреби, надсилається керівникові проєкту;

8.2 Життєвий цикл дефекту

Усі проблеми, виявлені під час тестування, будуть записані в документ Word.

Рисунок 8.1 ілюструє життєвий цикл дефекту в процесі тестування програмного забезпечення. Дефект проходить кілька етапів – від створення та підтвердження, через призначення розробнику, виправлення та перевірку до остаточного закриття. Якщо рішення не задовольняє QA, дефект може бути повторно відкрит для доопрацювання.

Рисунок В.12 – Тест-план сторінка 12

13

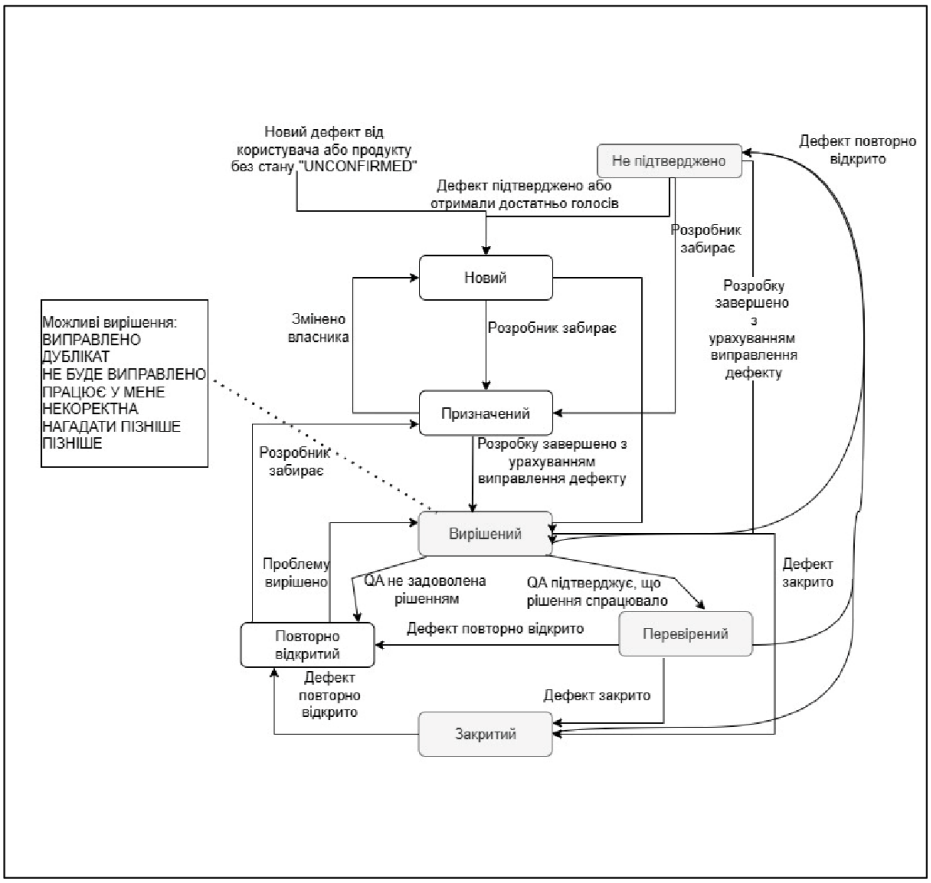



Рисунок 8.1 – Схема життєвого циклу дефекту

8.3 Типи тестування

У межах проєкту «Naviria» повинні застосовуватись декілька типів тестування, які забезпечують комплексну перевірку як функціональних, так і нефункціональних вимог до системи. Кожен з типів спрямований на виявлення дефектів на певному рівні реалізації або взаємодії компонентів системи.

Тестування «чорного ящика» (Black-box Testing) використовується для перевірки функціональності веб- та мобільного інтерфейсу з позиції кінцевого користувача, без доступу до внутрішньої логіки коду.

Рисунок В.13 – Тест-план сторінка 13

Тестувальник перевіряє відповідність фактичної поведінки системи очікуваним результатам згідно зі специфікаціями.

Тестування «білого» та «сірого» ящика (White-box та Gray-box Testing). Для API застосовується тестування з частковим або повним знанням внутрішньої логіки системи. Це дозволяє перевірити коректність реалізації контролерів, обробку вхідних параметрів, валідацію, бізнес-логіку та обробку виняткових ситуацій на рівні коду.

Інтеграційне тестування (Integration Testing) проводиться для перевірки взаємодії між різними компонентами системи: серверною частиною (API), базою даних MongoDB, фронтендом та мобільним клієнтом. Мета – виявити помилки, що виникають при обміні даними між модулями.

Функціональне тестування (Functional Testing) спрямоване на верифікацію реалізації функціональних вимог: автентифікації, керування цілями, досягненнями, сповіщеннями, соціальними взаємодіями тощо. Тестування виконується відповідно до тест-кейсів, побудованих на основі специфікацій і сценаріїв користування.

Системне тестування (System Testing) охоплює повністю інтегровану систему та перевіряє її відповідність встановленим вимогам у межах повноцінних наскрізних сценаріїв. Наприклад: створення облікового запису, додавання цілі, отримання сповіщення, взаємодія з іншими користувачами, чат з ШІ

Регресійне тестування (Regression Testing) проводиться після внесення змін у код або виправлення дефектів. Мета – переконатися, що нововведення не порушили стабільну роботу існуючого функціоналу. Включає автоматизовані та ручні перевірки критичних користувацьких сценаріїв.

Навантажувальне тестування (Performance Testing) використовується для перевірки поведінки системи в умовах високого навантаження. За

Рисунок В.14 – Тест-план сторінка 14

допомогою Apache JMeter тестується стійкість серверної частини й інтерфейсів при обробці великої кількості одночасних запитів, зокрема вимірюється час відповіді, швидкість обробки та стійкість до збоїв.

Приймальне тестування (User Acceptance Testing, UAT) проводиться на завершальному етапі перед випуском системи та виконується внутрішнім користувачем (представником команди), щоб оцінити загальну зручність, відповідність бізнес-вимогам і готовність системи до впровадження.

### 8.4 Визначення рівнів серйозності та пріоритетності дефектів

Поля серйозність (Severity) (див. табл. 8.1) та пріоритет (Priority) (див. табл. 8.2) є критично важливими для класифікації дефектів і визначення черговості їх усунення. Вони дозволяють ефективно організувати процес усунення помилок, враховуючи як технічну критичність, так і вплив на користувача або бізнес-цілі проєкту «Naviria».

Таблиця 8.1 – Класифікація серйозності дефектів

| ID | Рівень серйозності | Опис |
|---|---|---|
| 1 | Критична | Дефект призводить до аварійного завершення роботи системи, втрати даних, пошкодження БД або файлів, або блокує роботу всієї системи. Необхідне негайне усунення. |
| 2 | Висока | Основна функціональність недоступна або працює некоректно. Дефект серйозно впливає на користувача або блокує тестування інших модулів. Можливе існування складного або неінтуїтивного обходу. |

Рисунок В.15 – Тест-план сторінка 15

Кінець таблиці 8.1

| ID | Рівень серйозності | Опис |
|---|---|---|
| 3 | Середня | Некритичний дефект, який не впливає на основний функціонал, має простий обхідний шлях. Виникає у другорядних сценаріях або окремих модулях. |
| 4 | Низька | Орфографічні або візуальні помилки, недоліки в документації або вже підтверджені незначні дефекти, що не потребують термінового виправлення. |

Таблиця 8.2 – Класифікація пріоритетності дефектів

| ID | Рівень пріоритету | Опис |
|---|---|---|
| 1 | Обов’язково виправити | Дефект має бути усунутий негайно. Продукт не може бути переданий користувачам у наявності цієї помилки. |
| 2 | Бажано виправити | Дефект суттєвий та повинен бути усуненим у найкоротший термін. Її наявність може негативно вплинути на репутацію продукту. |
| 3 | Виправити за можливості | Виправлення не є критичним для поточного релізу, але бажане за наявності ресурсів. Якщо виправлення не впливає на строки випуску – виконати. |

Рисунок В.16 – Тест-план сторінка 16

Кінець таблиці 8.2

| ID | Рівень пріоритету | Опис |
|---|---|---|
| 4 | Не критично | Виправлення може бути відкладене на післярелізний період. Часто це незначні вдосконалення або додаткові функції, що виходять за межі поточних цілей. |

## 9. ПОТРЕБИ В РЕСУРСАХ ТА СЕРЕДОВИЩІ ТЕСТУВАННЯ

### 9.1 Інструменти тестування

У таблиці 9.1 наведено використання інструментів під час певних процесів.

Таблиця 9.1 – Таблиця процеси та їх інструменти

| Процес | Інструмент |
|---|---|
| Розробка ручних тест-кейсів | Microsoft Excel |
| Зберігання та відстеження тест-кейсів | Microsoft Excel |
| Виконання ручних тестів | Manual Testing, Excel |
| API-тестування | Swagger UI, Postman |
| Юніт-тестування (Unit testing) | NUnit, Moq |
| Автоматизоване UI-тестування (веб) | Cypress |
| Моніторинг навантаження та пам’яті | Android Studio |
| Навантажувальне тестування | Apache JMeter |
| Відстеження дефектів | Microsoft Word |
| Формування звітів про тестування | Microsoft Word, PDF |

Рисунок В.17 – Тест-план сторінка 17

18

9.2 Управління конфігурацією

Код СМ: Git

9.3 Тестове середовище

Для проведення тестування системи «Naviria» повинно бути використано мультиплатформне середовище, що охоплює сучасні браузери, емулятори, реальні мобільні пристрої та локальний сервер для запуску серверної частини застосунку.

- Операційне середовище:
  1) операційна система: Windows 11 Home (x64);
  2) браузери для вебтестування: Google Chrome (остання стабільна версія) та Microsoft Edge (остання стабільна версія).
- Мобільні пристрої для тестування:
  1) Samsung Galaxy A52 (Android 13).
- Серверне середовище (локальна машина):
  1) операційна система: Windows 11 Home;
  2) процесор: AMD Ryzen 7 5700U with Radeon Graphics, 1.80 GHz;
  3) оперативна пам’ять: 16 GB;
  4) диск: SSD 512 GB;
  5) мережеве підключення: локальна мережа;
  6) середовище запуску серверної частини: Visual Studio 2022;
  7) API-документація: Swagger UI локально.

Рисунок В.18 – Тест-план сторінка 18

10. ГРАФІК ТЕСТУВАННЯ

У таблиці 10.1 наведено графік тестування.

Таблиця 10.1 – Графік тестування

| Назва етапу | Початок | Завершення | Коментарі |
|---|---|---|---|
| Планування тестування | 08.04.2025 | 11.04.2025 | Організація процесу, уточнення обсягів |
| Аналіз вимог / Ознайомлення з документацією | 10.04.2025 | 16.04.2025 | Перегляд технічної документації |
| Перше розгортання до тестового середовища | 10.04.2025 | | Доступ до QA-середовища |
| Функціональне тестування ітерація 1 | 10.04.2025 | 21.04.2025 | Відповідно до переліку функціоналу |
| Розгортання ітерації 2 до QA-середовища | 21.04.2025 | | Нові зміни для перевірки |
| Функціональне тестування ітерація 2 | 21.04.2025 | 28.04.2025 | Розширене покриття функціоналу |
| Розгортання ітерації 3 до QA-середовища | 28.04.2025 | | |

Рисунок В.19 – Тест-план сторінка 19

Продовження таблиці 10.1

| Назва етапу | Початок | Завершення | Коментарі |
|---|---|---|---|
| Регресійне тестування | 28.04.2025 | 29.04.2025 | |
| Системне тестування | 29.04.2025 | 04.05.2025 | Тестування наскрізних сценаріїв |
| Функціональне тестування ітерація 3 | 29.04.2025 | 05.05.2025 | |
| Розгортання ітерації 4 до QA-середовища | 05.05.2025 | | |
| Системне тестування | 05.05.2025 | 07.05.2025 | |
| Функціональне тестування ітерація 4 | 05.05.2025 | 12.05.2025 | |
| Розгортання ітерації 5 до QA-середовища | 12.05.2025 | | |
| Регресійне тестування | 12.05.2025 | 14.05.2025 | |
| Системне тестування | 12.05.2025 | 19.05.2025 | |
| Функціональне тестування ітерація 5 | 14.05.2025 | 19.05.2025 | |

Рисунок В.20 – Тест-план сторінка 20

Кінець таблиці 10.1

| Назва етапу | Початок | | Завершення |
|---|---|---|---|
| Розгортання ітерації 6 до QA-середовища | 19.05.2025 | | |
| Регресійне тестування | 19.05.2025 | 20.05.2025 | |
| Функціональне тестування ітерація 6 | 20.05.2025 | 28.05.2025 | |
| Навантажувальне тестування | 25.05.2025 | 30.05.2025 | Apache JMeter |
| Системне тестування | 25.05.2025 | 30.05.2025 | |
| Приймальне тестування (UAT) | 26.05.2025 | 30.05.2025 | Внутрішнє тестування користувачами |
| Усунення критичних дефектів / фінальні перевірки | 27.05.2025 | 31.05.2025 | Перед стабільним релізом |

Рисунок В.21 – Тест-план сторінка 21

ТЕРМІНИ ТА АБРЕВІАТУРИ

| Термін / Абревіатура | Визначення |
|---|---|
| API | Application Programming Interface – інтерфейс програмування застосунків |
| QA | Quality Assurance – забезпечення якості |
| UAT | User Acceptance Testing – приймальне тестування |
| GUI | Graphical User Interface – графічний інтерфейс користувача |
| DB | Database – база даних |

Рисунок В.22 – Тест-план сторінка 22

# ДОДАТОК Г

## Результати навантажувального тестування

29.05.25, 15:04 Apache JMeter Dashboard

**Test and Report information**

| | |
|---|---|
| Source file | "results.jtl" |
| Start Time | "5/29/25, 2:52 PM" |
| End Time | "5/29/25, 2:55 PM" |
| Filter for display | "" |

**APDEX (Application Performance Index) (https://en.wikipedia.org/wiki/Apdex)**

| Apdex | T (Toleration threshold) | F (Frustration threshold) | Label |
|---|---|---|---|
| **0.042** | **500 ms** | **1 sec 500 ms** | **Total** |
| 0.042 | 500 ms | 1 sec 500 ms | HTTP Request |

**Requests Summary**

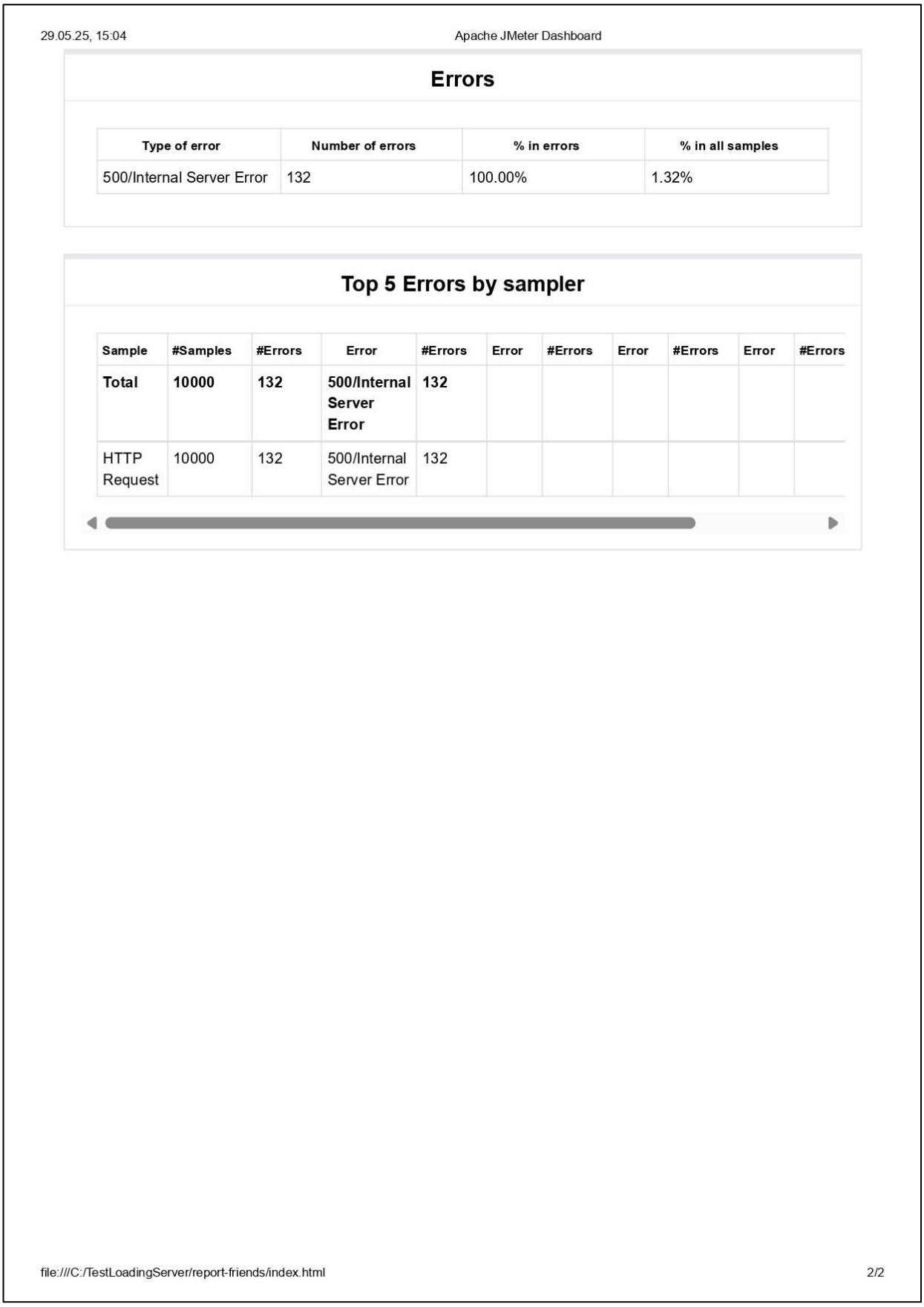


**Statistics**

| Requests | Executions | | | Response Times (ms) | | | | | | | Thr |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Label | #Samples | FAIL | Error % | Average | Min | Max | Median | 90th pct | 95th pct | 99th pct | Trans |
| **Total** | **10000** | **132** | **1.32%** | **15621.50** | **14** | **144733** | **15394.00** | **23747.50** | **29132.90** | **82229.15** | **55.03** |
| HTTP Request | 10000 | 132 | 1.32% | 15621.50 | 14 | 144733 | 15394.00 | 23747.50 | 29132.90 | 82229.15 | 55.03 |

file:///C:/TestLoadingServer/report-friends/index.html 1/2

Рисунок Г.1 – Звіт НТ на сервері для 1000 користувачів, сторінка 1



29.05.25, 15:04 Apache JMeter Dashboard

## Errors

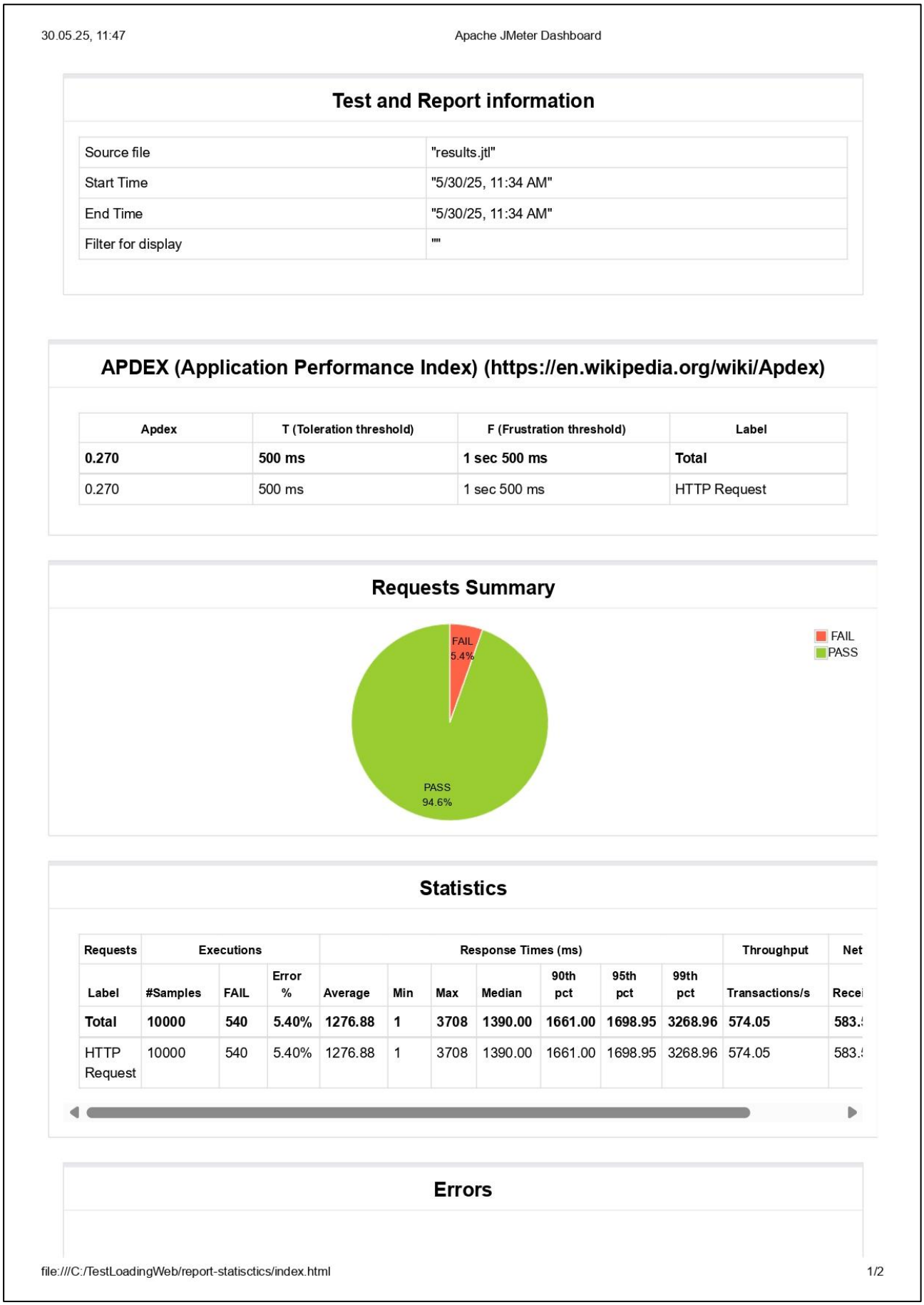
| Type of error | Number of errors | % in errors | % in all samples |
|---|---|---|---|
| 500/Internal Server Error | 132 | 100.00% | 1.32% |

## Top 5 Errors by sampler

| Sample | #Samples | #Errors | Error | #Errors | Error | #Errors | Error | #Errors | Error | #Errors |
|---|---|---|---|---|---|---|---|---|---|---|
| **Total** | **10000** | **132** | **500/Internal Server Error** | **132** | | | | | | |
| HTTP Request | 10000 | 132 | 500/Internal Server Error | 132 | | | | | | |

file:///C:/TestLoadingServer/report-friends/index.html 2/2

Рисунок Г.2 – Звіт НТ на сервері для 1000 користувачів, сторінка 2



30.05.25, 11:47 Apache JMeter Dashboard

## Test and Report information

| | |
|---|---|
| Source file | "results.jtl" |
| Start Time | "5/30/25, 11:34 AM" |
| End Time | "5/30/25, 11:34 AM" |
| Filter for display | "" |

## APDEX (Application Performance Index) (https://en.wikipedia.org/wiki/Apdex)

| Apdex | T (Toleration threshold) | F (Frustration threshold) | Label |
|---|---|---|---|
| **0.270** | **500 ms** | **1 sec 500 ms** | **Total** |
| 0.270 | 500 ms | 1 sec 500 ms | HTTP Request |

## Requests Summary



## Statistics

| Requests | Executions | | | Response Times (ms) | | | | | | | Throughput | Net |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Label | #Samples | FAIL | Error % | Average | Min | Max | Median | 90th pct | 95th pct | 99th pct | Transactions/s | Recei |
| **Total** | **10000** | **540** | **5.40%** | **1276.88** | **1** | **3708** | **1390.00** | **1661.00** | **1698.95** | **3268.96** | **574.05** | **583.** |
| HTTP Request | 10000 | 540 | 5.40% | 1276.88 | 1 | 3708 | 1390.00 | 1661.00 | 1698.95 | 3268.96 | 574.05 | 583. |

## Errors



Рисунок Г.3 – Звіт НТ на вебсторінці для 1000 користувачів, сторінка 1

| Type of error | Number of errors | % in errors | % in all samples |
|---|---|---|---|
| Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to localhost:5173 [localhost/127.0.0.1, localhost/0:0:0:0:0:0:0:1] failed: Connection refused: connect | 540 | 100.00% | 5.40% |

## Top 5 Errors by sampler

| Sample | #Samples | #Errors | Error | #Errors | Error | #Errors |
|---|---|---|---|---|---|---|
| **Total** | **10000** | **540** | **Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to localhost:5173 [localhost/127.0.0.1, localhost/0:0:0:0:0:0:0:1] failed: Connection refused: connect** | **540** | | |
| HTTP Request | 10000 | 540 | Non HTTP response code: org.apache.http.conn.HttpHostConnectException/Non HTTP response message: Connect to localhost:5173 [localhost/127.0.0.1, localhost/0:0:0:0:0:0:0:1] failed: Connection refused: connect | 540 | | |



Рисунок Г.4 – Звіт НТ на вебсторінці для 1000 користувачів, сторінка 2

# ДОДАТОК Д

## Тест-кейси

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC001 | Перехід на сторінку «Ком'юніті» | Запущений вебсайт, користувач авторизован | 1. Відкрити головну сторінку<br>2. Натиснути «Ком'юніті» у верхньому меню | Відкривається сторінка «Ком'юнітін» | Відкривається сторінка «Ком'юніті» | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC002 | Перевірка наявності блоку «Знайти друзів» | Сторінка «Ком'юніті», користувач авторизован | 1. Перейти на сторінку «Ком'юніті» | Блок з назвою «Знайти друзів» присутній на сторінці | Присутній | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC003 | Відображення списку потенційних друзів | Сторінка «Ком'юніті», користувач авторизован | 1. Перейти на сторінку «Ком'юнітін»<br>2. Обрати блок «Знайти друзів» | Відображається список користувачів для додавання у друзі | Відображається список користувачів для додавання у друзі | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC004 | Перевірка нікнейму у користувачів | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Перейти на сторінку<br>2. Перевірити, що біля кожного користувача є нікнейм | Біля кожного користувача відображається нікнейм | Біля кожного користувача відображається нікнейм | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC005 | Перевірка рівня користувачів | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Перейти на сторінку<br>2. Перевірити рівень користувача біля кожного елемента списку | Біля кожного користувача відображається рівень | Біля кожного користувача відображається рівень | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC006 | Перевірка наявності опису користувача | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Знайти користувача з описом<br>2. Перевірити наявність та зміст опису | Якщо опис є — він відображається | Якщо опис є — він відображається | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |

Рисунок Д.1 – Тест-кейси для тестування сторінки «Ком'юніті» через вебінтерфейс, частина 1

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC007 | Перевірка відсутності опису користувача | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Знайти користувача без опису<br>2. Перевірити текст біля користувача | Якщо опису немає — показується текст «Опис відсутній» | «Опис відсутній» | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC008 | Пошук за частиною нікнейму | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Ввести частину нікнейму у поле пошуку: gr<br>2. Натиснути кнопку «Пошук» | Виводяться користувачі, чиї нікнейми відповідають введеному тексту | Користувачі:<br>grom<br>tigrr | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC009 | Очистка поля пошуку | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Очистити поле пошуку<br>2. Натиснути кнопку «Пошук» | Відображається повний список користувачів для додавання в друзі | Відображається повний список користувачів для додавання в друзі | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC010 | Відправлення запиту в друзі | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Натиснути кнопку «Додати в друзі» біля користувача | Користувач зникає | Користувач зникає | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC011 | Фільтрація користувачів за категорією | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Відкрити список «Категорія»<br>2. Обрати «Література»<br>3. Натискнути кнопку «Пошук» | Виводяться користувачі, які створювали задачі у категорії «Література» | Користувачі:<br>yulia24<br>bennbarnes | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC012 | Фільтрація користувачів за категорією та пошук за нікнеймом | Користувач авторизован, сторінка «Ком'юніті», блок «Знайти друзів» | 1. Відкрити список «Категорія»<br>2. Обрати «Література»<br>3. Написати у поле для пошуку: yul<br>4. Натискнути кнопку «Пошук» | Виводяться користувачі, які створювали задачі у категорії «Література» та нікнейм містить | Користувачі:<br>yulia24 | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |

Рисунок Д.2 – Тест-кейси для тестування сторінки «Ком'юніті» через вебінтерфейс, частина 2

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC015 | Відображення категорій у випадаючому списку | Користувач авторизован, сторінка «Ком'юніті», блок «Запити» | 1. Натиснути на випадаючий список | Відображаються різні назви категорій | AI-generated tasks Література Спорт Навчання Благодійність Свята | Середній | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC016 | Відображення запитів користувачів у списку | У користувача є запити, блок «Запити» | 1. Переглянути список | У списку є запити з нікнеймом, рівнем, описом (якщо є) або написом «Опис відсутній» | У списку є запити з нікнеймом, рівнем, описом (якщо є) або написом «Опис відсутній» | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC017 | Перевірка кнопок «Відхилити» і «Прийняти» | У користувача є запити, блок «Запити» | 1. Перейти у блок "Запити" 2. Перевірити наявність кнопок «Відхилити» та «Прийняти» біля кожного запиту | Біля кожного запиту присутні обидві кнопки | Біля кожного запиту присутні обидві кнопки. Відхилити - червона. Прийняти - темно-синя | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC018 | Відхилення запиту | У користувача є запити, блок «Запити» | 1. Натиснути «Відхилити» біля одного із запитів | Запит зникає зі списку | Користувач зникає зі списку | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |
| TC019 | Прийняття запиту та оновлення списку друзів | У користувача є запити | 1. Натиснути «Прийняти» біля одного із запитів | Запит зникає зі списку і користувач з'являється у боковій панелі «Мої друзі» | Запит зникає зі списку і користувач з'являється у боковій панелі «Мої друзі» | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. Сервер: ASP.NET Core (.NET 8), Frontend: React | 29.05.2025 |

Рисунок Д.3 – Тестові кейси для блоку «Запити» на вебінтерфейсі

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC001 | Створення папки | Користувач авторизован, відкрито сторінку «Навірії» | 1. Натиснути на плюсик для додавання папки 2. Ввести назву папки: Нова папка 3. Натиснути кнопку «Зберегти» | Папка створена і відображається у списку папок | З'явилась «Нова папка» | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. | 29.05.2025 |
| TC002 | Створення задачі з підзадачами в папці | Створена папка «Нова папка» | 1. Натиснути на папку «Нова папка» | На екрані видно інформацію щодо папки, є кнопка-плючик «Створити нову задачу» | На екрані видно інформацію щодо папки, є кнопка-плючик «Створити нову задачу» | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. | 29.05.2025 |
| | | | 2. Натиснути на плюсик «Створити нову задачу» | Під написом «В процесі» виводиться форма для заповнення задачі | Під написом «В процесі» виводиться форма для заповнення задачі | | | | |
| | | | 3. У вікні створення задачі ввести назву «Почати грати на піаніно» | | | | | | |
| | | | 4. Ввести опис «Навчитись грати на піаніно» | | | | | | |
| | | | 5. Натиснути на випадаючий список з категоріями | Випадаючий список містить назви різних категорій | Випадаючий список містить назви різних категорій | | | | |
| | | | 6. Обрати категорію «Навчання» | | | | | | |
| | | | 7. Написати у полі тег «піаніно», натиснути плюсик для додавання тега | Тег збережено, з'явилось поле для нового тегу | Тег збережено, з'явилось поле для нового тегу | | | | |
| | | | 8. Натиснути на повзунок дедлайн | Відображення поля для вводу дати та значок календар | Відображення поля для вводу дати та значок календар | | | | |
| | | | 9. Натиснути на іконку календар та обрати дату з годинами 28.08.2025 20:22 | | | | | | |
| | | | 10. Написати цифру пріоритета: 1 | | | | | | |
| | | | 11. Обрати тип задачі: «З підзадачами» | Виводиться форма «Підзадачі» з номером 1 | Виводиться форма «Підзадачі» з номером 1 | | | | |
| | | | 12. Ввести назву «Вивчити ноти» | | | | | | |
| TC002 | Створення задачі з підзадачами в папці | Створена папка «Нова папка» | 13. Обрати тип підзадачі «Повторювана» | Виводиться список днів тижня | Виводиться список днів тижня | Високий | Pass | ОС: Windows 11, Браузер: Chrome 136.0.7103.114. | 29.05.2025 |
| | | | 14. Обрати Пн, Вт, Пт | Обрані дні змінили колір | Обрані дні змінили колір | | | | |
| | | | 15. Натиснути кнопку «Додати підзадачу» | Виводиться форма «Підзадачі» з номером 2 | Виводиться форма «Підзадачі» з номером 2 | | | | |
| | | | 16. У новій формі підзадачі ввести назву «Вивчити сонет» | | | | | | |
| | | | 17. Обрати тип підзадачі «Шкала» | Виводиться поле «Одиниця вимірювання» та «Ціль» | Виводиться поле «Одиниця вимірювання» та «Ціль» | | | | |
| | | | 18. Заповнити поле: Одиниця вимірювання -- «сонети» | | | | | | |
| | | | 19. Заповнити поле: Ціль -- 5 | | | | | | |
| | | | 20. Натиснути кнопку «Додати підзадачу» | Виводиться форма «Підзадачі» з номером 3 | | | | | |
| | | | 21. Ввести назву «Знайти вчителя» | | | | | | |
| | | | 22. Обрати тип підзадачі «Звичайна» | | | | | | |
| | | | 23. Натиснути кнопку «Зберегти» | На сторінці під написом Під написом «В процесі» видно створену задачу | На сторінці під написом Під написом «В процесі» видно «Почати грати на піаніно» | | | | |

Рисунок Д.4 – Тестові кейси для сторінки «Навірії» на вебінтерфейсі

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC_MOBILE_001 | Авторизація користувача та перевірка сторінки профілю | Мобільний застосунок запустили, емулятор активний | 1. Запустити мобільний додаток<br>2. Ввести логін: maria.kovalenko@example.com<br>3. Ввести пароль: Maria1234<br>4. Натиснути кнопку «Увійти»<br>5. Перевірити, що відкрилася сторінка «Ком'юніті»<br>6. Натиснути на аватарку в правому верхньому кутку<br>7. Перевірити, що відкрилася сторінка «Мій профіль»<br>8. Переконатися, що відображаються:<br>- нікнейм користувача<br>- аватарка<br>- шкала прогресу з рівнем<br>- кількість XP та залишок до наступного рівня<br>9. Перевірити наявність секції «Опис»<br>10. Перевірити наявність «Досягнень» і їх список<br>11. Проскролити вниз до списку друзів<br>12. Перевірити, що відображається список друзів і їх нікнейми | Користувач успішно авторизується, бачить свій профіль з усіма елементами | усі елементи були присутні, додаток працює стабільно | Високий | Pass | Medium Phone API 35 | 15.05.2025 |
| TC_MOBILE_002 | Перевірка валідації при зміні нікнейму (більше 22 символів) | Авторизований користувач, відкрито сторінку «Мій профіль» | 1. На сторінці «Мій профіль» натиснути на іконку олівця<br>2. У полі «Нікнейм» ввести значення довжиною понад 22 символи<br>3. Натиснути кнопку «Зберегти зміни» | Застосунок не оновлює профіль. Показано повідомлення про помилку валідації | Повідомлення, що нікнейм не може бути більше 20, зміни не збережені | Середній | Pass | Medium Phone API 35 | 15.05.2025 |
| TC_MOBILE_003 | Перевірка валідації при зміні пошти на не валідний (без "@") | Авторизований користувач, відкрито сторінку «Мій профіль» | 1. На сторінці профілю натиснути на іконку олівця<br>2. У полі «Пошта» ввести значення без символу @<br>3. Натиснути кнопку «Зберегти зміни» | Застосунок не оновлює профіль. Показано повідомлення про помилку валідації | Показано повідомлення про не валідний формат пошти | Середній | Pass | Medium Phone API 35 | 15.05.2025 |

Рисунок Д.5 – Тестові кейси для редагування профілю через мобільний застосунок

| ID | Опис | Умови | Кроки виконання тестів | Очікуваний результат | Фактичний результат | Пріоритет | Результат тесту | Оточення | Дата створення |
|---|---|---|---|---|---|---|---|---|---|
| TC_MOBILE_010 | Перевірка, що користувач отримує бали після натискання кнопки «Подарунок» | Користувач авторизований (maria992). У профілі є досягнення, за які ще не отримано бали. На сторінці «Досягнення» | 1. У MongoDB знайти користувача і перевірити значення поля points (наприклад: 288)<br>2. Натиснути кнопку «Подарунок» під досягненням «Good start»<br>3. Перевірити, що кнопка змінила колір і напис змінився на «Отримано»<br>4. Повторно перевірити поле points в MongoDB -- значення має збільшитись на 50 (очікуване: 338) | Кнопка «Подарунок» стає неактивною, текст змінюється на «Отримано», у БД значення points збільшено | Для користувача 680cc31a9c376cfd07332ef0. Значення в БД: 338; кнопка змінила стан і текст | Високий | Pass | Medium Phone API 35 | 30.05.2025 |
| TC_MOBILE_015 | Відмітка задачі як виконано у правильний день (п'ятниця) | Користувач авторизований . Є задача типу «Повторювана (вт, пт)» зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Пт» (сьогодні п'ятниця) | Виводиться повідомлення: «Відмітка успішна!», день активовано | Повідомлення з'явилось, день | Високий | Pass | Samsung Galaxy A52 | 30.05.2025 |
| TC_MOBILE_016 | Перевірка, що не можна відмітити день, який не входить до розкладу задачі | Користувач авторизований . Є задача типу «Повторювана (вт, пт)» зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Нд» (не є визначеним днем) | Нічого не відбувається, кнопка сірого кольору, повідомлення не з'являється | Кнопка неактивна, реакції немає | Середній | Pass | Samsung Galaxy A52 | 30.05.2025 |
| TC_MOBILE_017 | Перевірка, що не можна поставити відмітку на майбутній день | Користувач авторизований . Є задача типу «Повторювана (вт, пт, сб)» зі статусом «У процесі» | 1. Перейти на сторінку «Завдання»<br>2. Відкрити задачу зі статусом «У процесі»<br>3. Відкрити відповідну задачу<br>4. Натиснути на кнопку «Сб», сьогодні п'ятниця | Виводиться повідомлення: «День ще не настав» | Виводиться повідомлення: «День ще не настав» | Середній | Pass | Samsung Galaxy A52 | 30.05.2025 |

Рисунок Д.6 – Тестові кейси для відмічання дню в задачах через мобільний застосунок

# ДОДАТОК Е

## Аналіз продуктивності застосунку


Рисунок Е.1 – Результат «View Live Telemetry»

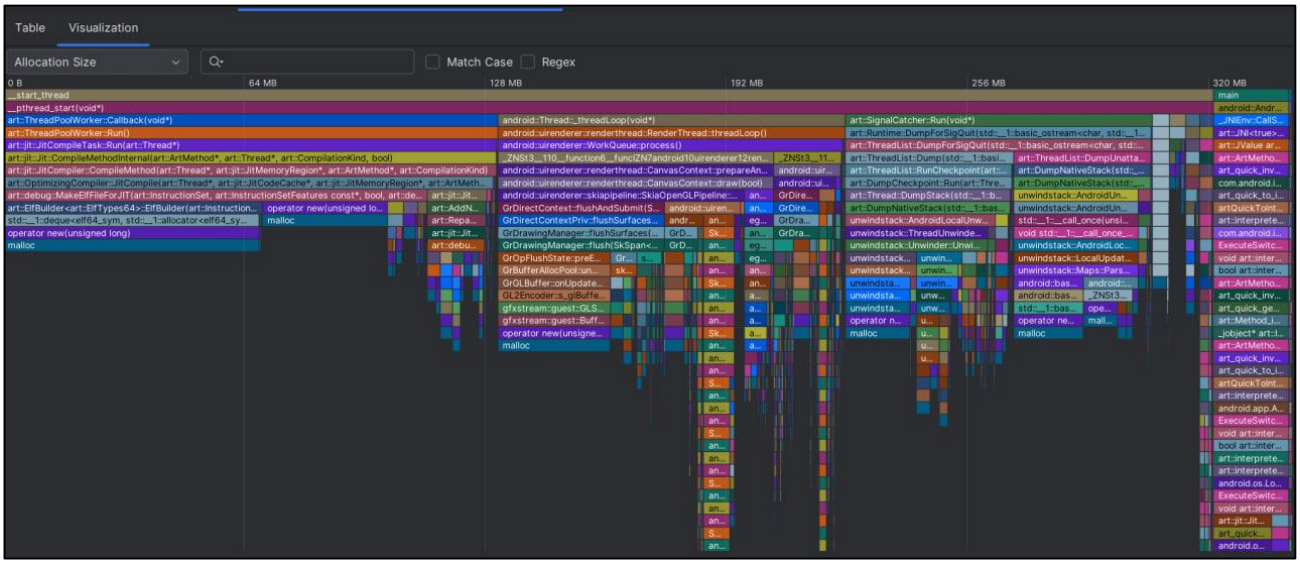



Рисунок Е.2 – Візуалізація моніторингу споживання пам’яті

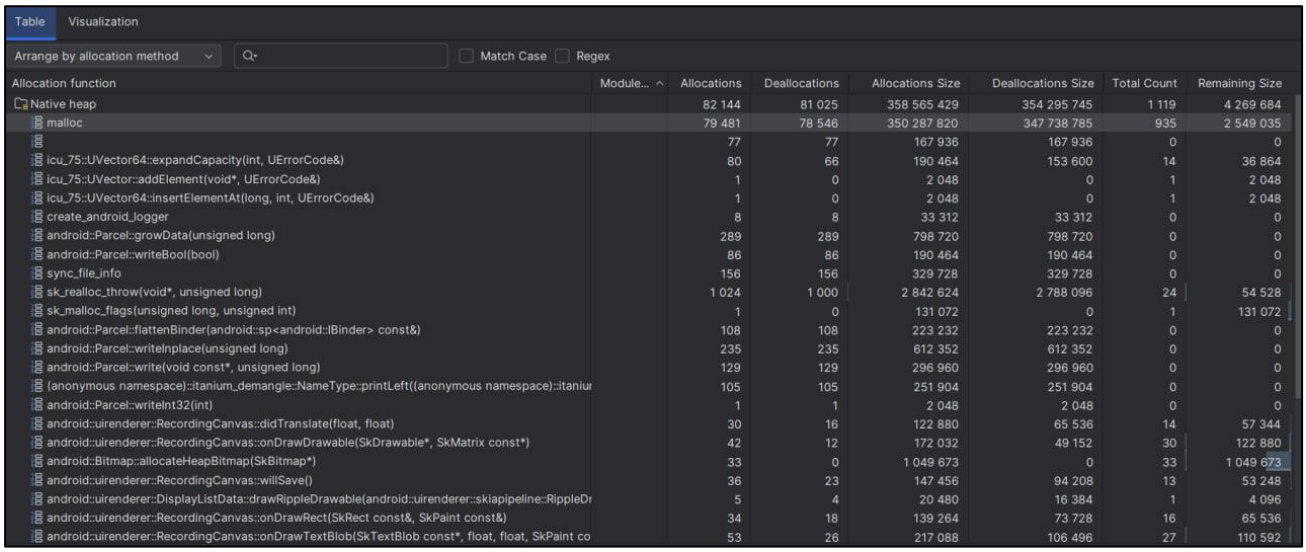


Table | Visualization

Arrange by allocation method | Match Case | Regex

| Allocation function | Module... | Allocations | Deallocations | Allocations Size | Deallocations Size | Total Count | Remaining Size |
|---|---|---|---|---|---|---|---|
| Native heap | | 82 144 | 81 025 | 358 565 429 | 354 295 745 | 1 119 | 4 269 684 |
| malloc | | 79 481 | 78 546 | 350 287 820 | 347 738 785 | 935 | 2 549 035 |
| | | 77 | 77 | 167 936 | 167 936 | 0 | 0 |
| icu_75::UVector64::expandCapacity(int, UErrorCode&) | | 80 | 66 | 190 464 | 153 600 | 14 | 36 864 |
| icu_75::UVector::addElement(void*, UErrorCode&) | | 1 | 0 | 2 048 | 0 | 1 | 2 048 |
| icu_75::UVector64::insertElementAt(long, int, UErrorCode&) | | 1 | 0 | 2 048 | 0 | 1 | 2 048 |
| create_android_logger | | 8 | 8 | 33 312 | 33 312 | 0 | 0 |
| android::Parcel::growData(unsigned long) | | 289 | 289 | 798 720 | 798 720 | 0 | 0 |
| android::Parcel::writeBool(bool) | | 86 | 86 | 190 464 | 190 464 | 0 | 0 |
| sync_file_info | | 156 | 156 | 329 728 | 329 728 | 0 | 0 |
| sk_realloc_throw(void*, unsigned long) | | 1 024 | 1 000 | 2 842 624 | 2 788 096 | 24 | 54 528 |
| sk_malloc_flags(unsigned long, unsigned int) | | 1 | 0 | 131 072 | 0 | 1 | 131 072 |
| android::Parcel::flattenBinder(android::sp<android::IBinder> const&) | | 108 | 108 | 223 232 | 223 232 | 0 | 0 |
| android::Parcel::writeInplace(unsigned long) | | 235 | 235 | 612 352 | 612 352 | 0 | 0 |
| android::Parcel::write(void const*, unsigned long) | | 129 | 129 | 296 960 | 296 960 | 0 | 0 |
| (anonymous namespace)::itanium_demangle::NameType::printLeft((anonymous namespace)::itaniu | | 105 | 105 | 251 904 | 251 904 | 0 | 0 |
| android::Parcel::writeInt32(int) | | 1 | 1 | 2 048 | 2 048 | 0 | 0 |
| android::uirenderer::RecordingCanvas::didTranslate(float, float) | | 30 | 16 | 122 880 | 65 536 | 14 | 57 344 |
| android::uirenderer::RecordingCanvas::onDrawDrawable(SkDrawable*, SkMatrix const*) | | 42 | 12 | 172 032 | 49 152 | 30 | 122 880 |
| android::Bitmap::allocateHeapBitmap(SkBitmap*) | | 33 | 0 | 1 049 673 | 0 | 33 | 1 049 673 |
| android::uirenderer::RecordingCanvas::willSave() | | 36 | 23 | 147 456 | 94 208 | 13 | 53 248 |
| android::uirenderer::DisplayListData::drawRippleDrawable(android::uirenderer::skiapipeline::RippleDr | | 5 | 4 | 20 480 | 16 384 | 1 | 4 096 |
| android::uirenderer::RecordingCanvas::onDrawRect(SkRect const&, SkPaint const&) | | 34 | 18 | 139 264 | 73 728 | 16 | 65 536 |
| android::uirenderer::RecordingCanvas::onDrawTextBlob(SkTextBlob const*, float, float, SkPaint co | | 53 | 26 | 217 088 | 106 496 | 27 | 110 592 |

Рисунок Е.3 – Таблиця моніторингу споживання пам’яті