

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Комп'ютерних наук і технологій
(повне найменування факультету)

Комп'ютерні системи та мережі
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалаврський
(ступінь вищої освіти)

на тему : РОЗРОБКА СИСТЕМИ ТЕСТУВАННЯ СОЦІАЛЬНОЇ
ПЛАТФОРМИ GOSEECHAT

Виконав(ла) : студент(ка) 4 курсу,
групи КНТ-522

Спеціальності

123 Комп'ютерна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ТЕРЕНТЬЄВ О.С.
(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.
(ПРИЗВИЩЕ та ініціали)

Рецензент ТЯГУНОВ Д.М.
(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра комп'ютерних систем та мереж
Ступінь вищої освіти бакалаврський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри КУДЕРМЕТОВ Р.К.

«14» квітня 2026 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ТЕРЕНТЬЄВА Олексія Сергійовича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка системи тестування соціальної платформи GoSeeChat

керівник проєкту (роботи) к.т.н., доцент, ТЯГУНОВА М.Ю.,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 14 » квітня 2026 року № 151

2. Строк подання студентом проєкту (роботи) 01.06.2026 р.

3. Вихідні дані до проєкту (роботи) опис предметної області соціальної платформи GoSeeChat, специфікація бізнес-вимог, інструментарій інженерного аналізу мережеских запитів Chrome DevTools та Swagger UI, середовище баг-трекінгу та управління завданнями Atlassian Jira, процес розробки системи тестування

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз предметної області ;

2) Проєктування та інструментальне забезпечення тестування платформи;

3) Практичне впровадження системи тестування ;

4) Аналіз отриманих результатів ;

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М.Ю.		
Нормоконтроль	ГОЛУБ Т.В.		

7. Дата видачі завдання «14» квітня 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області	до 18.04.2026	
2	Теоретичні основи ручного тестування	до 22.04.2026	
3	Проектування системи	до 24.04.2026	
4	Інструментальне забезпечення тестування	до 28.04.2026	
5	Проектування тестових сценаріїв та техніки тест-дизайну для обраних ролей.	до 01.05.2026	
6	Тестування функцій ролі Продюсера	до 05.05.2026	
7	Тестування функцій ролі Креатора	до 12.05.2026	
8	Тестування функцій ролі Клієнт	до 17.05.2026	
8	Аналіз виявлених дефектів, регресія та статистика виправлень	до 22.05.2026	
9	Оформлення пояснювальної записки	до 25.05.2026	
10	Проходження нормоконтролю	до 27.05.2026	
11	Перевірка на наявність академічного плагіату	до 28.05.2026	
12	Проходження рецензування	до 01.06.2026	

Студент(ка)

(підпис)

Олексій ТЕРЕНТЬЄВ

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Марія ТЯГУНОВА

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
82с., 11 табл., 24 рис., 15 джерел.

CHROME DEVTOOLS, GOSEECHAT, MANUAL QA, SWAGGER UI,
ВАЛІДАЦІЯ ДАНИХ, КЛІЄНТ-СЕРВЕРНА АРХІТЕКТУРА, КРЕАТОР,
МОДЕЛЮВАННЯ ТЕСТУВАННЯ, ПРОДЮСЕР

Об'єкт розробки – система тестування соціальної платформи GoSeeChat.

Мета роботи – підвищення загальної стабільності функціонування, гарантування високого рівня фінансової безпеки, забезпечення ідемпотентності транзакційних модулів та комплексна оптимізація існуючого процесу інструментальної верифікації наскрізної клієнт-серверної бізнес-логіки розподіленої архітектури соціальної платформи GoSeeChat за рахунок впровадження системи тестування.

У першому розділі проаналізовано бізнес-вимоги (BRS), архітектуру та рольову модель (Клієнт, Креатор, Продюсер) платформи GoSeeChat.

У другому розділі обґрунтовано методологію ручного тестування та оптимізовано техніки тест-дизайну для максимального покриття.

У третьому розділі сформовано стратегію, а також визначено методику контролю REST API та трафіку через Swagger UI й Chrome DevTools, проведено повноцінне тестування системи.

У четвертому розділі був проведений повний аналіз результатів тестування системи, знайдених багів, їх оформлення у системі трекінгу Jira.

ЗМІСТ

Скорочення та умовні позначки	5
Вступ.....	6
1.Аналіз предметної області.....	9
1.1. Методологія ручного тестування	15
1.2. Специфіка QA для соціальних платформ та приватності даних	17
1.3. Організація тестового середовища : Dev- та Prod-сервери	20
2. Проєктування та інструментальне забезпечення тестування платформи	25
2.1. Загальна структура та алгоритм функціонування платформи.....	27
2.2. Вибір та обґрунтування інструментів Manual QA	30
2.3. Стратегія тестування та матриця покриття вимог	34
2.4. Проєктування тестових сценаріїв та техніки тест-дизайну для обраних ролей.....	37
3. Практичне впровадження системи тестування.....	42
3.1. Тестування функціональних модулів за ролями	44
3.1.1. Тестування функцій ролі Продюсера.....	46
3.1.2. Тестування функцій ролі Креатора.....	52
3.1.3. Тестування функцій ролі Клієнт	59
3.2. Процес тестування в умовах розподілених середовищ (Dev/Prod)	66
4. Аналіз отриманих результатів.....	70
4.1. Аналіз виявлених дефектів, регресія та статистика виправлень	70
Висновки.....	76
Перелік джерел посилання.....	78
Додаток А Слайди презентації.....	80

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

API	— Програмний інтерфейс додатка
BRS	— Специфікація бізнес-вимог
Dev-сервер	— Середовище розробки (Development server)
G-pass	— Внутрішня розрахункова одиниця платформи
GUI	— Графічний інтерфейс користувача (Graphical User Interface)
Manual QA	— Ручне забезпечення якості ПЗ (Manual Quality Assurance)
OAuth	— Протокол відкритої авторизації (Facebook, Google, Apple)
Prod-сервер	— Робоче середовище системи (Production server)
ID	— Унікальний ідентифікатор об'єкта (Identifier)
RTM	— Матриця простежуваності вимог
UI	— Користувацький інтерфейс (User Interface)
UX	— Користувацький досвід (User Experience)

ВСТУП

Глобальні соціальні медіа стрімко трансформуються зі звичайних засобів комунікації у багатофункціональні високотехнологічні екосистеми. Сучасні платформи інтегрують функціонал соціальних мереж, цифрових маркетплейсів та платіжних сервісів. Це дозволяє користувачам безпосередньо монетизувати інтелектуальну власність і власні професійні навички. Відповідно, технічна якість програмного забезпечення (ПЗ) напряду визначає ринковий успіх продукту. Будь-які дефекти в бізнес-логіці призводять до критичних репутаційних та економічних втрат компанії.

Необхідність гарантування високої надійності та безпеки даних у нішевих соціальних сервісах, які оперують конфіденційною інформацією та фінансовими потоками, визначає актуальність дослідження. Об'єктом аналізу є соціальна платформа GoSeeChat. Ця система має багаторівневу рольову модель, яка розділяє користувачів на три категорії: клієнти (споживачі), креатори (автори контенту) та продюсери (керівники команд). Головна технічна особливість платформи полягає в глибокій інтеграції фінансових інструментів. Система використовує внутрішню розрахункову одиницю G-pass (еквівалент 1 євро) та складні алгоритми автоматичного розподілу прибутків усередині продюсерських команд. Найменші помилки в тарифікації повідомлень, розрахунку часток доходу чи функціонуванні приватних режимів (зокрема, опції «Hide My Media») генерують критичні ризики для безпеки користувачів та стабільності бізнесу.

Метою роботи є підвищення загальної стабільності функціонування, гарантування високого рівня фінансової безпеки, забезпечення ідемпотентності транзакційних модулів та комплексна оптимізація існуючого процесу інструментальної верифікації наскрізної клієнт-серверної бізнес-логіки розподіленої архітектури соціальної платформи GoSeeChat за рахунок впровадження системи тестування [1].

Для досягнення поставленої мети необхідно вирішити такі завдання:

- провести глибокий аналіз предметної області та технічної специфікації бізнес-вимог GoSeeChat для ідентифікації архітектурних ризиків; теоретичне обґрунтування та вибір технік тест-дизайну для оптимізації тестового покриття;
- спроектувати наскрізну стратегію тестування та формування структури тестової документації для кабінетів Клієнта, Креатора та Продюсера;
- реалізувати інструментальний контроль REST API та мережевого трафіку за допомогою засобів Swagger UI та Chrome DevTools для ранньої локалізації дефектів;
- провести комплексне тестування критичних компонентів системи, включаючи фінансовий модуль Wallet, систему мікротранзакцій, приватні режими контенту та логіку взаємоблокування; а також моніторинг життєвого циклу аномалій у середовищі Atlassian Jira, проведення регресійних випробувань після рефакторингу коду та фінальна статистична оцінка стабільності системи.

Суть дослідження [2] полягатиме в проектуванні та впровадженні комплексної системи верифікації, здатної повністю охопити графічний інтерфейс користувача (GUI) та наскрізну бізнес-логіку взаємодії клієнтської і серверної частин. Специфіка сучасних соціальних сервісів унеможлиблює використання виключно автоматизованих рішень. Валідація нелінійних сценаріїв вимагає застосування методів ручного тестування (Manual QA). До таких сценаріїв належать: верифікація профілів (KYC), модерація медіаконтенту відповідно до стандартів «18+» та «Sensitive», а також контроль виконання угод про рівень сервісу (SLA) щодо оперативності відповідей авторів. Ручне тестування забезпечує гнучкість, необхідну для імітації реальної поведінки користувачів за умов динамічних бізнес-вимог. Це критично для GoSeeChat, де черговість видачі контенту та алгоритми розсилок (Newsletter) залежать від багатьох перемінних параметрів.

Технічна специфікація бізнес-вимог (BRS) GoSeeChat послужить основою для розробки стратегії тестування. Цей документ містить вичерпний опис функціональних модулів, ієрархії прав доступу та архітектурної побудови системи. Інженери використовували як вихідні дані вимоги до клієнт-серверної

взаємодії через API, дизайн-макети інтерфейсів під різні роздільні здатності екранів (від мобільних 390px до десктопних 1920px), а також правила валідації вхідних даних у формах реєстрації, користувацьких профілях і розділі Help Center.

Висока технічна складність внутрішніх інтеграційних процесів платформи зумовлює потребу в комплексному контролі якості. Застосування інструментів Swagger UI для валідації API-запитів та Chrome DevTools для діагностики фронтенду й аналізу мережевого трафіку дозволить локалізувати дефекти на ранніх етапах життєвого циклу розробки. Такий підхід дозволить оптимізувати QA-процеси в розподіленій інфраструктурі, де первинне тестування нових функцій відбувається на Dev-сервері, а фінальна приймальна перевірка цілісності системи — на Prod-середовищі.

У межах дослідження [3] буде реалізовано наскрізну стратегію ручного тестування, що базується на аналізі предметної області та декомпозиції рольової моделі на окремі об'єкти перевірки. Практична частина охоплюватиме проєктування структури тестової документації в середовищі Google Workspace та формування тест-кейсів за допомогою технік тест-дизайну: еквівалентного поділу та аналізу граничних значень. Особливу увагу буде приділено верифікації критичних компонентів: фінансового модуля Wallet, системи мікротранзакцій, інструментів збору донатів (Tip Box) та ролі продюсера для координації команд. Заключний етап роботи охоплюватиме моніторинг життєвого циклу виявлених дефектів, проведення регресійного тестування після внесення виправлень та фінальну оцінку ефективності розробленої системи забезпечення якості.

Наукова та практична цінність отриманих результатів полягатиме у формуванні структурованої інженерної методики контролю якості для медіаплатформ з інтегрованою системою мікротранзакцій. Впровадження цієї методики буде мінімізувати ризики збоїв у алгоритмах тарифікації та нарахування коштів, підвищує рівень захисту конфіденційних даних користувачів і покращує показники задоволеності кінцевих споживачів (User Experience) під час взаємодії з платформою.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Соціальна платформа GoSeeChat [4] функціонує за принципом клієнт-серверної архітектури та є складним програмним комплексом, орієнтованим на надання послуг з обміну цифровим контентом. На відміну від стандартних комунікаційних сервісів, цей об'єкт розробки інтегрує елементи фінансової платформи (FinTech) та системи управління контентом (CMS).

Проектувальник заклав у систему архітектуру кросплатформного вебрішення з високим рівнем адаптивності. Згідно з технічною специфікацією бізнес-вимог (BRS), графічний інтерфейс користувача (GUI) має стабільно функціонувати у діапазоні роздільних здатностей від мобільних пристроїв (390x844 px) до повнорозмірних десктопних моніторів (1920x1080 px). Така динаміка суттєво ускладнює процеси верифікації верстки. Елементи керування (наприклад, нижній Bottom Navigation Bar у мобільній версії або стаціонарний Sidebar у десктопній конфігурації) динамічно змінюють просторове положення та функціональну доступність залежно від параметрів екрана.

Архітектурна логіка системи чітко розмежовує користувацький досвід на дві основні зони:

- неавторизована зона: розроблена для ознайомлення відвідувачів із можливостями платформи. Робоча область має максимально великі розміри. Тут діє стрічковий режим відображення даних, який містить блоки Creators (мінімум 10 профілів, відібраних модераторами) та NewsFeed (вертикальна стрічка публікацій з аватарами та медіаконтентом). Будь-яка спроба інтерактивної взаємодії (перегляд відео, відкриття постів) викликає модальне поп-ап вікно з вимогою реєстрації;

- авторизована зона: суттєво розширює навігаційні можливості користувача. Хедер стає стаціонарним елементом, забезпечуючи швидкий перехід до розділів Creators, NewsFeed, Videos, Photos та Streams. Особливої уваги під час тестування вимагає блок «My Messages», розташований ліворуч від основного

контенту. Він розділений на вкладки GoSee (для обробки нових контактів) та Contacts (для діалогів із платними взаємодіями), що є ключовим технічним індикатором монетизації платформи.

Для детальної діагностики взаємодії системних компонентів виконано функціональну декомпозицію платформи на три базові модулі, взаємодію яких наведено на рисунку 1.1.



Рисунок 1.1 — Схема взаємодії функціональних модулів Mediabank, Wallet та Newsletter

Модуль управління медіаданими (Mediabank) – це хмарне сховище, де кожен Креатор зберігає, редагує та категоризує фото-, відео- та аудіофайли. Технічна особливість полягає у впровадженні логіки виключення повторного використання контенту. Система візуально маркує файли рамками різних кольорів залежно від їхнього поточного стану: New (синя рамка), In Queue (жовта), Active (зелена) або Banned (червона).

Фінансовий модуль (Wallet) забезпечує оперування внутрішньою валютою G-pass, де 1 одиниця номіналу еквівалентна 1 євро. Для QA-інженера цей модуль

має найвищий пріоритет перевірки через складну логіку тарифікації. Інженер верифікує «баблову» тарифікацію (повідомлення до 280 символів об'єднуються в один платний блок протягом 60 секунд, а довші тексти від 281 до 4000 символів тарифікуються окремо), купівлю медіасетів Клієнтами та механізм збору пожертв Тір Вох («банка») з фіксацією прогресу в реальному часі за встановленої мінімальної цілі в 10 G-pass.

Сервіс автоматизації розсилок (Newsletter Engine) - це система ретаргетингу, яка динамічно сегментує Клієнтів на групи за рівнем залученості: COLD (категорії Intro, Topics) та WARM (категорії Fans, Posts) з подальшою генерацією повідомлень типу Follow-up. Верифікація модуля потребує тестування математичних алгоритмів «Rockstar», які підвищують пріоритет видачі Креатора в системі на основі його активності та швидкості відповіді на запити Клієнтів (Response Time).

Рольова модель GoSeeChat побудована на суворій ієрархії, що визначає права доступу до конфіденційних даних та фінансових інструментів.

Процес інтеграції нового користувача з роллю Клієнта (Client) вимагає обов'язкового вибору інтересів (Topics). Система пропонує 5 тематичних груп (включаючи Auto/Sport, Fashion/Beauty, Travelling/Business тощо), і вибір хоча б однієї категорії є технічною умовою для завершення реєстрації. Клієнт має право на поповнення балансу, споживання цифрового контенту та налаштування приватності профілю за допомогою функції «Hide My Media».

Центральною бізнес-роллю є Креатор (Creator) — власник цифрового контенту. Для активації можливостей монетизації Креатор обов'язково проходить процедуру верифікації особистості (KYC) через завантаження фото документів та відеоідентифікацію, а також виконує нормативи первинного наповнення сховища: завантажує мінімум 5 фото, 5 відео та публікує 1 пост. Креатор оперує інструментом автоматичного привітання (Chat Greeting) та самостійно встановлює індивідуальні прайси на контент.

Управлінську та фінансову структуру системи контролює Продюсер (Producer). Продюсер налаштовує Таблицю розподілу доходу, де діє суворе

математичне правило «100% на рядок». Роль Креатора у цій фінансовій схемі виступає буфером: будь-яке збільшення відсоткової частки Продюсера автоматично зменшує безпосередню частку Креатора від транзакцій, при цьому значення доходу Креатора технічно не може бути меншим за нуль.

Порівняльну характеристику розподілу правових повноважень між трьома основними ролями наведено в таблиці 1.1.

Таблиця 1.1 — Порівняльна характеристика прав доступу ролей системи GoSeeChat

Функціонал	Продюсер (Producer)	Креатор (Creator)	Клієнт (Client)
Редагування профілю	Ні	Так	Так
Перегляд фінансової звітності	Так	Так (лише власних)	Так (лише балансу)
Управління розсилками (Newsletter)	Так	Так	Ні
Виведення коштів (Payout)	Так	Так	Ні

Окрім рольової диференціації, архітектура системи передбачає обмеження можливостей користувачів залежно від рівня акаунта. Параметри лімітів систематизовано в таблиці 1.2.

Таблиця 1.2 — Системні ліміти та можливості для різних рівнів акаунтів

Функціональні параметри	Рівень Standard	Рівень Premium Lvl 2	Рівень Premium Lvl 3
Кількість публікацій на добу	3	10	Безліміт
Об'єм сховища Mediabank	40	80	160
Кількість активних нотаток (Notes)	8	16	64

Для забезпечення безпеки даних та модерації контенту платформа GoSeeChat використовує гібридну модель захисту. Автоматична фільтрація базується на базі даних заборонених слів (Banned Words) та модулі унікалізації тексту (Check text for duplicates). Цей алгоритм автоматично порівнює поля «About Me» та «My Motto» Креаторів з інформацією інших користувачів для запобігання плагіату та поширення автоматизованого спаму.

Ручна модерація передбачає накопичувальну систему штрафних балів. При виявленні некоректно маркованого контенту категорій «18+» або «Sensitive», модератори нараховують профілю Креатора штрафні бали. При накопиченні 10 балів система автоматично блокує акаунт користувача.

Важливим інженерним вузлом є Центр підтримки (Help Center), архітектура якого складається з 8 категорій допомоги. У цей модуль інтегровано спеціалізовану форму відповідно до вимог Акта про цифрові послуги ЄС (Digital Services Act — DSA) для репортування протиправного контенту. Валідація форм у Help Center включає обов'язкову автоматичну перевірку доменів електронної пошти за верифікованим переліком урядових адрес для обробки офіційних юридичних запитів від правоохоронних органів.

У підсумку, GoSeeChat являє собою фінансово-комунікаційну інфраструктуру з жорсткою бізнес-логікою. Наявність таких модулів, як Newsletter Engine з сегментацією на COLD/WARM групи та алгоритмів Rockstar для пріоритезації активних авторів, вимагає від тестувальника не лише перевірки графічного інтерфейсу, а й глибокої верифікації математичних та логічних моделей, закладених у бекенд системи.

1.1 Методологія ручного тестування

Методологія ручного забезпечення якості (Manual Quality Assurance) у межах комп'ютерної інженерії є цілісною інженерною дисципліною. Вона

передбачає верифікацію відповідності програмного продукту встановленим бізнес-вимогам та технічним специфікаціям. Для архітектури соціальної платформи GoSeeChat ручне тестування є обов'язковим компонентом контролю якості. Воно дозволяє комплексно оцінити користувацький досвід та валідувати логіку бізнес-процесів, які складно автоматизувати через нелінійні сценарії взаємодії між Клієнтом, Креатором та Продюсером .

Життєвий цикл тестування програмного забезпечення інтегрується в загальний процес розробки та починається з аналізу специфікації бізнес-вимог (BRS). На етапі аналізу вимог інженер з якості вивчає логіку функціонування модуля тарифікації повідомлень, де вартість транзакції змінюється залежно від довжини тексту (межа 280 символів) та часу активної сесії взаємодії Клієнта з Креатором. Отримані дані формують фундамент для планування тестування: визначається обсяг робіт, обирається інструментарій та розробляється стратегія верифікації фінансових транзакцій у внутрішній валюті G-pass, еквівалентній 1 Євро.

Процес проєктування тестів включає розробку детальних тестових сценаріїв на основі конкретних параметрів системи, зафіксованих у BRS. Сюди належать ліміти завантаження медіафайлів Креатором у хмарне сховище Mediabank та правила валідації унікальності нікнеймів користувачів. Інженер виконує тестування в розподілених середовищах: на Dev-сервері здійснює локалізацію первинних дефектів, а на Prod-сервері проводить фінальне приймальне тестування для підтвердження цілісності платформи. Завершення циклу тестування супроводжується аналізом виявлених дефектів та оцінкою готовності продукту до релізу .

Взаємозв'язку рівнів тестування та життєвого циклу розробки платформи наведено на рисунку 1.2.

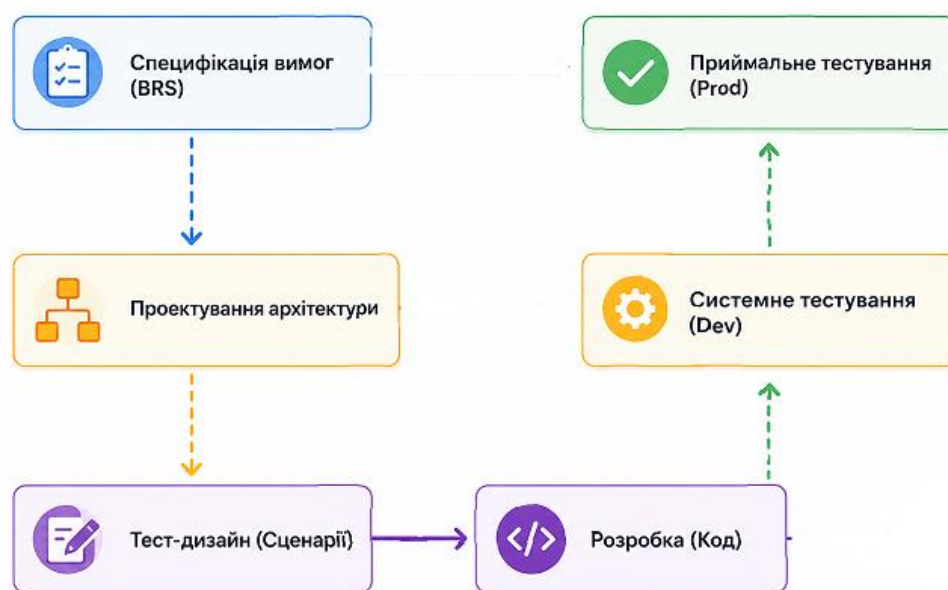


Рисунок 1.2 — Схема взаємозв'язку рівнів тестування та життєвого циклу розробки GoSeeChat

Функціональне тестування платформи охоплює перевірку коректності роботи всіх модулів згідно з технічним завданням. Воно включає верифікацію інтерфейсу Продюсера, де ключовим елементом є таблиця розподілу часток доходу, який генерують Креатори під час взаємодії з Клієнтами. Особливу увагу інженер приділяє фінансовому модулю Wallet, оскільки помилки в алгоритмах нарахування відсотків G-pass призводять до прямих фінансових втрат учасників платформи.

Паралельно проводиться тестування графічного інтерфейсу (GUI) та адаптивності, що є технічно складним завданням через необхідність підтримки широкого діапазону роздільних здатностей від 390px до 1920px. Кожен елемент, від мобільного Bottom Navigation Bar до десктопного Sidebar, має зберігати функціональну доступність при масштабуванні вікна браузера.

Тестування зручності використання (Usability) та безпеки займає чітке місце в загальній методології. Соціально-фінансовий характер платформи вимагає інтуїтивно зрозумілої навігації та високого рівня приватності. Верифікації підлягають механізми обмеження доступу, зокрема функція Креатора «Hide My

Media», яка приховує контент від Клієнтів, що не входять до списку контактів. Також інженер перевіряє систему автоматичної модерації за списками Banned Words та логіку блокування акаунтів Креаторів при накопиченні 10 штрафних балів за некоректне маркування контенту категорій «18+» або «Sensitive».

Ефективність ручного тестування підвищується завдяки застосуванню науково обґрунтованих технік тест-дизайну:

- Еквівалентний поділ: використовується для групування вхідних даних у формах реєстрації Клієнтів та Креаторів.
- Аналіз граничних значень: дозволяє протестувати критичні точки системи, такі як текстові повідомлення довжиною рівно 280 та 281 символ, або ліміти поповнення балансу гаманця Клієнтом.

Застосування цих технік мінімізує кількість тест-кейсів за умови збереження максимального покриття вимог.

Важливим завершальним етапом кожного ітераційного циклу є регресійне тестування. Його мета полягає в підтвердженні того, що виправлення виявлених дефектів або впровадження нових функцій (наприклад, алгоритму Rockstar для пріоритезації Креаторів із найкращим Response Time) не порушило стабільність вже існуючого функціонала системи.

У підсумку, методологія ручного тестування для проєкту GoSeeChat є комплексним системним підходом, який поєднує класичні практики QA з глибокою технічною специфікацією соціальних FinTech-платформ та чітким розмежуванням прав Продюсера, Креатора й Клієнта.

1.2 Специфіка QA для соціальних платформ та приватності даних

Проектування стратегії тестування для соціальної платформи GoSeeChat вимагає від інженера із забезпечення якості глибокого розуміння специфічних ризиків, притаманних системам із суміщенням соціальної взаємодії, фінансових

операцій (FinTech) та хостингу користувацького контенту. Специфіка QA-процесів у цьому контексті полягає у верифікації багаторівневих прав доступу та забезпеченні цілісності конфіденційної інформації в умовах динамічного оновлення даних.

Забезпечення якості соціальної платформи передбачає перевірку механізмів контролю доступу, які базуються виключно на взаємодії ролей Клієнта, Креатора та Продюсера. Центральним об'єктом верифікації виступає функція Креатора «Hide My Media», яка обмежує видимість цифрового контенту для Клієнтів, що не входять до його списку контактів.

Процес тестування цієї функції включає перевірку ініціалізації станів системи. QA-інженер підтверджує, що при активації відповідного тумблера у профілі Креатора, бекенд-сервер коректно генерує заблюрені прев'ю-зображення та захищає прямі посилання на оригінальні медіафайли за допомогою тимчасових токенів. Це унеможливорює несанкціонований доступ Клієнтів до контенту через прямий URL.

Окремим завданням для тестувальника є верифікація маскування персональних даних (PII) в інтерфейсах ролі Продюсера. Згідно з вимогами безпеки та стандартами GDPR, Продюсер не повинен мати доступу до повних електронних адрес або платіжних реквізитів Клієнтів та Креаторів. Інженер перевіряє коректність роботи алгоритмів маскування (наприклад, відображення лише перших та останніх символів Email), що дозволяє запобігти внутрішнім витокам інформації.

Платформа GoSeeChat оперує значними обсягами мультимедійних даних, що потребує впровадження гібридної системи модерації, контроль над якою з боку менеджменту здійснює Продюсер. QA-інженер верифікує логіку обробки контенту за категоріями «Zero Tolerance», «18+» та «Sensitive». Тестувальник моделює сценарії, за яких завантаження Креатором контенту з нульовою толерантністю призводить до миттєвого блокування акаунта без права на автоматичне відновлення доступу.

Особливу увагу інженер приділяє тестуванню системи штрафних санкцій.

Необхідно валідувати алгоритм, за яким накопичення 10 штрафних балів за некоректне маркування контенту категорії «18+» автоматично переводить акаунт Креатора у статус «Banned». Це потребує перевірки цілісності бази даних аудиту, де система фіксує кожну дію з боку Продюсера, причину відхилення контенту та його ідентифікатор.

Паралельно верифікації підлягає інтегрована форма Акта про цифрові послуги ЄС (Digital Services Act — DSA). Інженер перевіряє, як система валідує скарги від Клієнтів та юридичні запити, контролюючи домени пошти за списками верифікованих урядових адрес (government-approved).

Для активації інструментів монетизації Креатор та Продюсер проходять обов'язковий етап KYC (Know Your Customer). QA-специфікація охоплює перевірку складних форм реєстрації, де користувач завантажує фото документів (ID) та проходить відеоверифікацію із промовлянням контрольної фрази. Тестувальник перевіряє всі переходи станів цієї процедури: від статусу «In Progress» до «Approved» або «Rejected» із обов'язковим контролем відправки сервісних Email-повідомлень Клієнтам та Креаторам із зазначенням конкретної причини відмови.

Важливим інженерним завданням є тестування механізму «Check text for duplicates». Система порівнює текстові поля «About Me» та «My Motto» нового Креатора з уже існуючими записами в БД для виявлення фейкових профілів. QA-інженер верифікує точність пошуку за комбінаціями «Ім'я + Прізвище + Дата народження» або «Країна + Прізвище», підтверджуючи, що потенційні дублікати коректно підсвічуються в інтерфейсі Продюсера для подальшого аналізу анкети.

Впровадження внутрішньої валюти G-pass (1 G-pass = 1 Євро) накладає на тестувальника обов'язок верифікувати точність розрахунків та ізоляцію фінансових даних:

- креатор повинен мати доступ виключно до власної статистики доходів та балансу гаманця;
- продюсер володіє доступом до загальної таблиці розподілу доходу.

QA-інженер перевіряє математичну логіку автоматичного перерахунку

часток у фінансовій таблиці Продюсера, де діє суворе правило «100% на рядок». Будь-яке збільшення відсоткової частки Продюсера має автоматично списуватися з частки Креатора, при цьому система повинна блокувати спроби встановлення значень, за яких дохід Креатора стає меншим за нуль.

Додатково тестуванню підлягають механізми модуля Wallet, а саме ліміти на суми поповнення балансу Клієнтом (мінімум 20, максимум 1000 G-pass) та валідація платіжних методів. Специфічним аспектом є перевірка обробки помилок при недостатньому балансі: система має миттєво переривати платний чат між Клієнтом та Креатором або блокувати доступ до преміальних повідомлень, відображаючи модальне вікно для поповнення рахунку Клієнта.

1.3 Організація тестового середовища : Dev- та Prod-сервери

Ефективність забезпечення якості в межах комп'ютерної інженерії безпосередньо залежить від архітектури та конфігурації тестового середовища. Для таких багатокомпонентних систем, як соціальна платформа GoSeeChat, де взаємодіють фінансові модулі (Wallet), системи мобільної та десктопної адаптивності графічного інтерфейсу користувача (GUI) та складні рольові ієрархії, критично важливим є розмежування середовищ розробки та промислової експлуатації. Такий підхід мінімізує ризики виникнення дефектів у робочому середовищі та підтримує ізоляцію процесів верифікації нових функцій.

Організація інфраструктури проєкту базується на використанні двох основних типів серверів: Dev-сервера (Development/Staging) та Prod-сервера (Production). Кожне з цих середовищ має власні технічні параметри, ізольовані бази даних та конфігураційні налаштування програмних інтерфейсів (API). Це вимагає від QA-інженера чіткого розмежування тестових даних та методів верифікації для кожного контуру.

Dev-сервер виступає ізольованим середовищем для програмістів та інженерів із забезпечення якості. Саме тут проводиться первинне тестування функціональних модулів одразу після завершення їх розробки. Основна мета функціонування цього середовища полягає у виявленні первинних дефектів: грубих логічних помилок, проблем із відображенням GUI на різних роздільних здатностях (від 390px до 1920px) або некоректної роботи API-запитів, які інженер валідує через Swagger UI [6].

У контексті платформи GoSeeChat на Dev-сервері проходять верифікацію такі аспекти бізнес-логіки:

- логіка тарифікації: перевірка коректності списання внутрішньої валюти G-pass за повідомлення довжиною понад 280 символів або за доступ Клієнтів до медіасетів Креатора у сховищі Mediabank;
- рольові переходи та взаємодія: верифікація сценаріїв фінансового підпорядкування, коли Креатор підключається до ролі Продюсера для розподілу отриманого доходу;
- інтеграція підсистеми розсилок (Newsletter Engine): тестування алгоритмів сегментації Клієнтів на групи COLD та WARM без ризику надсилання тестових повідомлень реальним користувачам платформи.

Цикл розгортання програмного коду та послідовність етапів його тестування наведено на рисунку 1.3.

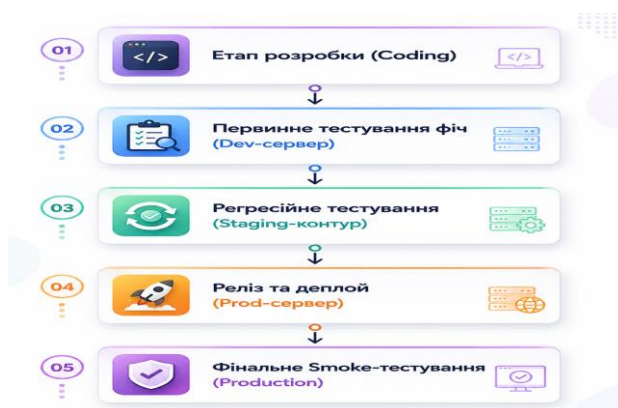


Рисунок 1.3 — Схема потоку розгортання та верифікації програмного забезпечення між Dev- та Prod-серверами

Prod-сервер — це робоче промислове середовище системи, де перебувають реальні Клієнти, Креатори та Продюсери, а також здійснюються справжні фінансові транзакції. Тестування на цьому рівні має виключно приймальний характер (Acceptance / Smoke Testing). Основна задача QA-інженера на Prod-сервері — підтвердити цілісність системи після оновлення та переконатися, що нова функціональність коректно синхронізувалася з існуючою базою даних та зовнішніми сервісами.

Особливості верифікації на Prod-сервері охоплюють:

- використання реальних конфігурацій: перевірка OAuth-авторизації Клієнтів та Креаторів через Google, Apple ID та Facebook із використанням валідних продуктивних токенів та сертифікатів безпеки.
- фінансова безпека: верифікація роботи модуля Wallet з реальними платіжними шлюзами, де 1 G-pass чітко відповідає номіналу в 1 Євро.
- верифікація процедури KYC: перевірка процесів завантаження Креаторами фото документів та відеоідентифікації в реальних умовах взаємодії з інтерфейсом Продюсера.

Порівняльну характеристику параметрів обох середовищ наведено в таблиці 1.3.

Для фахівця з комп'ютерної інженерії важливо контролювати ідентичність конфігурацій обох серверів, щоб запобігти виникненню дефектів, пов'язаних із різницею в оточенні. Це досягається шляхом верифікації наступних технічних параметрів:

- версіонування API: використання Swagger UI дозволяє QA-інженеру порівнювати структури відповідей сервера на обох середовищах, забезпечуючи зворотну сумісність.
- синхронізація баз даних: тестування міграцій БД, особливо в частині таблиць розподілу доходу Продюсера, де суворе правило «100% на рядок» має виконуватися незалежно від обраного контуру.
- адаптивність та кешування: використання інструментів Chrome DevTools для аналізу мережевої активності (Network) та очищення кешу після деплою нових версій фронтенду, що критично для складних елементів графічного

інтерфейсу користувача (зокрема, нижнього мобільного Bottom Navigation Bar або модальних вікон обрізки зображень «Crop Media» Креатора).

Таблиця 1.3 — Порівняльна характеристика Dev- та Prod-середовищ для тестувальника

Параметр порівняння	Середовище розробки (Dev-сервер)	Промислове середовище (Prod-сервер)
Мета проведення тестування	Пошук, локалізація та виправлення дефектів	Приймальний контроль якості (Acceptance)
Рівень стабільності контуру	Низький (через постійне оновлення коду)	Високий (допускає лише релізні збірки)
Рівень технічних ризиків	Мінімальний (ізолювана інфраструктура)	Максимальний (ризик фінансових втрат)
Основний інструментарій QA	Swagger UI, Postman, Chrome DevTools	Chrome DevTools (Network), логування

Процес тестування в умовах розподілених середовищ є ітераційним. Після успішного завершення повного регресійного циклу на Dev-сервері, інженер готує зміни до випуску, фіксує результати в матриці покриття вимог (RTM) та погоджує реліз. Одразу після розгортання (деплою) коду на Prod-сервері проводиться фінальна перевірка критичного шляху (Critical Path Testing). Вона гарантує, що оновлення не порушило базовий користувацький досвід: можливість реєстрації Клієнта, перехід Креатора до бізнес-ролі та механізми поповнення фінансового балансу.

Розмежування інфраструктури на Dev- та Prod-сервери є фундаментальною вимогою для забезпечення якості соціальної платформи GoSeeChat . Dev-середовище підтримує безпечну зону для глибокої верифікації бізнес-логіки та адаптивності інтерфейсу, тоді як Prod-середовище гарантує фінальну

працездатність продукту для кінцевого споживача. Такий інженерний підхід дозволяє локалізувати дефекти на ранніх етапах розробки, підтримуючи фінансову та репутаційну стабільність програмного комплексу в процесі його життєвого циклу.

2 ПРОЄКТУАННЯ ТА ІНСТРУМЕНТАЛЬНЕ ЗАБЕЗПЕЧЕННЯ ТЕСТУВАННЯ ПЛАТФОРМИ

Сутність розробки комплексної системи тестування для сучасних високонавантажених медіаплатформ полягає не просто у механічній перевірці працездатності окремих елементів інтерфейсу, а у створенні цілісного, науково обґрунтованого інженерного комплексу верифікації та валідації. У комп'ютерній інженерії система тестування розглядається як структурована сукупність взаємопов'язаних методів, нормативних стандартів, технік тест-дизайну, інструментальних засобів та тестових артефактів, що функціонують як єдиний механізм контролю якості програмного забезпечення на всіх етапах його життєвого циклу.

При проєктуванні системи забезпечення якості для платформи GoSeeChat сутність розробки розкривається через декілька фундаментальних аспектів.

По-перше, це створення формалізованої стратегії покриття складної бізнес-логіки. Оскільки платформа оперує багаторівневою рольовою моделлю (Клієнт, Креатор, Продюсер) та інтегрованим фінансовим ядром Wallet, розробка системи тестування полягає в декомпозиції цих великих модулів на ізольовані, але взаємопов'язані об'єкти перевірки. Система має гарантувати, що будь-яка нелінійна дія користувача — від тарифікації повідомлень у внутрішній валюті G-pass до активації приватних режимів контенту (зокрема, «Hide My Media») — буде оброблена сервером ідентично до вимог технічної специфікації (BRS).

По-друге, сутність системи полягає в інтеграції методів чорної скриньки

(Black Box) та сірої скриньки (Grey Box). Якщо класичне тестування інтерфейсу обмежується візуальним контролем GUI, то розроблена інженерна система поєднує поведінковий аналіз із низькорівневою валідацією клієнт-серверних запитів. Це передбачає проектування таких сценаріїв, де паралельно з діями на фронтенді здійснюється моніторинг структури JSON-пакетів, перевірка JWT-токенів авторизації на рівні API Gateway та аналіз асинхронного трафіку.

По-третє, архітектурна сутність розробки полягає в оптимізації тестових наборів за допомогою математичних технік тест-дизайну. Створення системи тестування вимагає уникнення надмірності перевірок при збереженні максимальної ефективності. Застосування методів еквівалентного поділу та аналізу граничних значень дозволяє згрупувати нескінченну кількість вхідних параметрів (наприклад, сум транзакцій, лімітів виводу коштів чи полів реєстрації) у чіткі, репрезентативні класи, мінімізуючи часові витрати на виконання тестів.

У підсумку, сутність системи тестування полягає у забезпеченні стійкості розподіленої інфраструктури застосунку. Проектована система створює надійний бар'єр для дефектів під час їхнього мігрування між тестовим (Dev) та промисловим (Prod) середовищами, організує прозорий контроль життєвого циклу помилок і впроваджує методологію регресійного контролю, що в сукупності мінімізує репутаційні та фінансові ризики вебплатформи.

2.1 Загальна структура та алгоритм функціонування платформи

Проектний етап розробки системи забезпечення якості вимагає чіткого формального опису топології та внутрішньої логіки взаємодії компонентів об'єкта тестування. Соціальна платформа GoSeeChat розгорнута за класичною триланковою архітектурною моделлю, де кожен рівень функціонує як ізольований шар із регламентованими інтерфейсами взаємодії. Організаційну схему рівнів

системи та розподіл між клієнтською і серверною частинами представлено на рисунку 2.1.

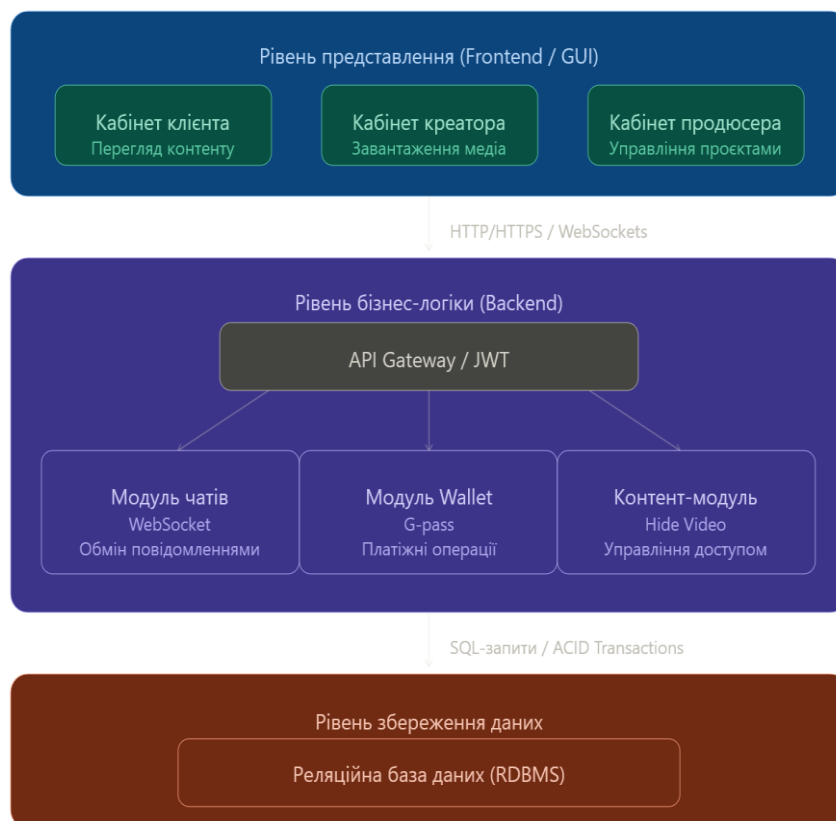


Рисунок 2.1 - Організаційна схема рівнів системи

Функціональний розподіл архітектурних шарів визначає межі верифікації для QA-інженера:

- рівень представлення (Presentation Tier / Frontend): реалізований за допомогою бібліотеки React, виконує рендеринг інтерфейсу користувача (GUI), динамічно адаптує відображення під різні типи пристроїв та здійснює первинну клієнтську валідацію форм;

- рівень бізнес-логіки (Application Tier / Backend): містить центральні модулі обробки даних, включаючи транзакційне ядро гаманця Wallet (оперування внутрішніми одиницями G-pass), сервіси асинхронних чатів, логіку обробки приватного контенту та матрицю прав доступу користувачів;

– рівень збереження даних (Data Tier / Database): реляційна СУБД, що здійснює довготривале збереження інформації, підтримує цілісність даних та обробляє транзакції з дотриманням вимог ACID;

Згідно з технічною специфікацією бізнес-вимог (BRS), внутрішній алгоритм функціонування системи під час обробки користувацьких запитів та транзакцій базується на покроковій верифікації станів [8]. Послідовність обробки типової події на платформі — від ініціації дії користувачем у GUI до фіксації змін у базі даних — наведено на блок-схемі (рисунок 2.2).

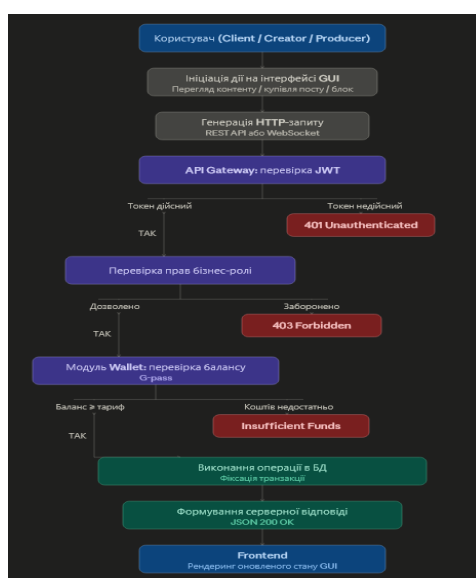


Рисунок 2.2 - Послідовність обробки типової події на платформі

Алгоритм передбачає, що будь-яка маніпуляція контентом або балансом ініціює відправку асинхронного запиту до бекенд-частини. API Gateway перевіряє сесійний токен (JWT) на валідність та актуальність прав доступу конкретної ролі. У разі успішного проходження авторизації запит передається до бізнес-модулів (Wallet, Чат або Контентний блок), де виконуються низькорівневі перевірки даних. Після успішного виконання транзакції на рівні бази даних сервер повертає статус-код 200 OK разом із відповідним JSON-пакетом, що слугує тригером для фронтенду щодо миттєвого перерендеру елементів інтерфейсу користувача .

Формалізація архітектурної структури та алгоритмічних зв'язків платформи дозволяє чітко розмежувати зони відповідальності під час розробки тестових сценаріїв, комбінуючи методи Black Box тестування (на рівні Presentation Tier) та Grey Box тестування (на рівні Application Tier).

2.2 Вибір та обґрунтування інструментів Manual QA

Ефективність стратегії забезпечення якості для соціальної платформи GoSeeChat залежить від вибору інструментарію, що здатний забезпечити повну прозорість взаємодії між розподіленими компонентами системи. Оскільки об'єкт розробки базується на складній клієнт-серверній архітектурі, де фронтенд-частина (реалізована на базі бібліотеки React) взаємодіє з бекенд-сервером через REST API та протокол WebSocket, вибір засобів тестування охоплює всі рівні передачі та обробки даних. Теоретичною основою вибору інструментів Manual QA у цій роботі є необхідність верифікації графічного інтерфейсу користувача (GUI), внутрішньої бізнес-логіки, фінансової цілісності транзакцій та безпеки персональних даних.

Для верифікації серверної логіки та коректності роботи програмного інтерфейсу додатка (API) основним інструментом обрано Swagger UI. З точки зору системної інженерії, Swagger є інтерактивною документацією на базі специфікації OpenAPI, яка дозволяє тестувальнику виконувати низькорівневі перевірки безпосередньо на рівні HTTP-запитів.

У проєкті GoSeeChat використання Swagger UI обумовлено потребою в ізольованому тестуванні модулів, які мають складні алгоритми обробки даних:

- модуль фінансових транзакцій (Wallet): оскільки платформа оперує внутрішньою валютою G-pass (де 1 одиниця номіналу дорівнює 1 Євро), інженер обов'язково перевіряє математичну точність та ідемпотентність операцій. Swagger дозволяє валідувати структури JSON-відповідей, де кожна покупка медіасету або

оплата повідомлення містить обов'язкове булеве поле `isPaid` та коректну ціну. Це дозволяє виявити помилки тарифікації до їхнього відображення в інтерфейсі користувача.

- система авторизації та розмежування прав доступу: платформа підтримує три ключові ролі (Клієнт, Креатор та Продюсер), кожна з яких має унікальний набір повноважень. Перевірка механізму Bearer Token у Swagger UI дозволяє QA-інженеру імітувати сесії різних користувачів, контролюючи безпеку ендпоінтів та запобігаючи несанкціонованому доступу Клієнтів до приватних медіафайлів у Mediabank Креатора.

- валідація вхідних даних: згідно з BRS, система має суворі правила валідації параметрів (наприклад, довжина нікнейма Клієнта від 5 до 24 символів або фільтрація за списком стоп-слів). Swagger дозволяє проводити негативне тестування шляхом передачі некоректних типів даних для підтвердження того, що сервер повертає правильні коди помилок (400 Bad Request, 403 Forbidden).

Послідовність взаємодії тестувальника з серверною частиною через інтерактивний інтерфейс наведено на рисунку 2.3.

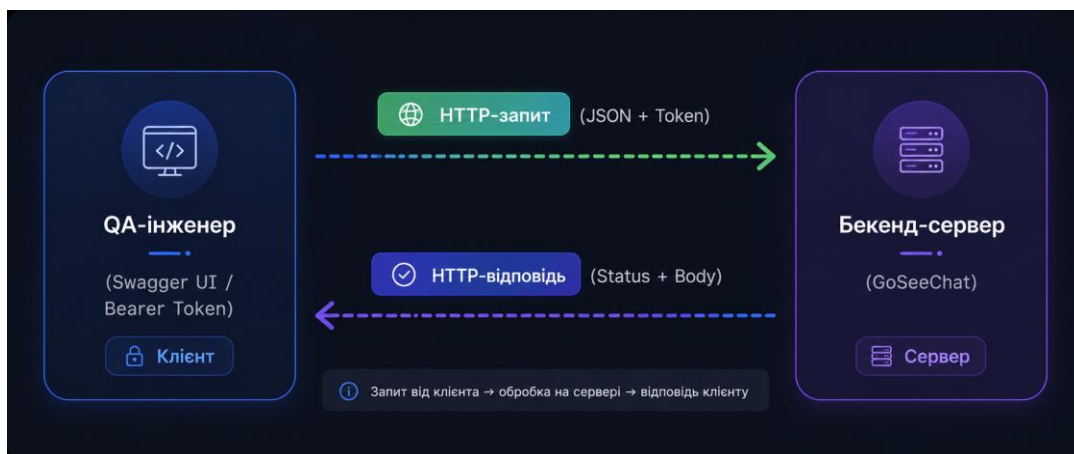


Рисунок 2.3 — Схема взаємодії тестувальника з бекендом GoSeeChat через Swagger UI

Для забезпечення якості користувацького досвіду (UX) та стабільності графічного інтерфейсу невід'ємним інструментом є Chrome DevTools [7]. Його

теоретична значущість полягає в можливості аналізувати стан вебдодатка безпосередньо в середовищі виконання — браузері.

Основними функціональними зонами використання DevTools у межах проєкту є:

- аналіз мережевої активності (Панель Network): інженер застосовує цей інструмент для тестування підсистеми розсилок (Newsletter Engine) та чатів у реальному часі. QA-візуалізація дозволяє відстежувати час завантаження медіаконтенту, аналізувати Payload запитів та виявляти затримки в сокет-з'єднаннях, які забезпечують роботу індикатора набору тексту під час комунікації Клієнта з Креатором.

- тестування адаптивності та сумісності верстки: додаток GoSeeChat має стабільно підтримувати роздільні здатності від 390x844 px (мобільна версія) до 1920x1080 px (десктопні монітори). Використання режиму Device Mode дозволяє верифікувати коректність трансформації інтерфейсних елементів (таких як нижній Bottom Navigation Bar або висувне меню Клієнта) за різних умов доступу.

- діагностика стану сховищ (Панель Application): верифікація механізмів зберігання даних на стороні клієнта (Cookies, Local Storage) є необхідною для перевірки збереження токенів сесії та мовних налаштувань. Наприклад, при зміні мови інтерфейсу в хедері обране значення має автоматично записуватися в локальне сховище браузера, що й підтверджує цей інструмент.

Технічні критерії, що зумовили вибір конкретного інструментарію для забезпечення якості системи, узагальнено в таблиці 2.1.

Окрім засобів прямої технічної діагностики, стратегія тестування включає використання хмарних сервісів Google Workspace для ведення тестової документації та Figma для візуальної верифікації. Figma виступає в ролі еталонної моделі, за якою інженер з якості перевіряє відповідність розмірів шрифтів, колірних статусів контенту в Mediabank (наприклад, червона рамка для забаних Продюсером файлів Креатора) та ієрархії відступів вебверстки.

Таблиця 2.1 — Технічні критерії вибору інструментів тестування для GoSeeChat

Інструмент тестування	Об'єкт безпосередньої верифікації	Технічна перевага інструмента
Swagger UI	Серверна бізнес-логіка, REST API, фінансові ендпоінти	Ізольоване тестування HTTP-запитів, валідація JSON-схем, підміна Bearer Token
Chrome DevTools	Графічний інтерфейс (GUI), мережева активність, клієнтські сховища	Аналіз WebSocket-трафіку, емуляція роздільних здатностей екрана, моніторинг Local Storage
Figma	Дизайн-макети, еталонна інтерактивна модель	Контроль відповідності інтерфейсу специфікаціям (Pixel-perfect), перевірка колірних статусів

Теоретичний аналіз обраного інструментарію підтверджує його достатність для реалізації мети роботи. Комбінація Swagger UI та Chrome DevTools дозволяє реалізувати системний підхід до тестування GoSeeChat, охоплюючи всі етапи життєвого циклу розробки: від перевірки первинних вимог на Dev-сервері до фінального приймального тестування на Prod-сервері. Використання цих професійних засобів підтримує високу точність локалізації дефектів, мінімізує ризики фінансових втрат у системі оперування G-pass та гарантує надійність захисту персональних даних Клієнтів, Креаторів і Продюсерів.

2.3 Стратегія тестування та матриця покриття вимог

Проектування стратегії тестування для соціальної платформи GoSeeChat є етапом забезпечення якості, який визначає загальний підхід, набір інструментів та пріоритетність перевірок у межах життєвого циклу розробки програмного

забезпечення. У контексті комп'ютерної інженерії стратегія розглядається як високорівневий документ, що інтегрує технічні вимоги специфікації BRS з інженерними методами верифікації клієнт-серверної взаємодії. Основна мета стратегії полягає у мінімізації ризиків виникнення критичних дефектів у модулях, які оперують фінансовими транзакціями у валюті G-pass та персональними даними користувачів.

Стратегія охоплює всі рівні архітектури платформи, розподіляючи зусилля між функціональними та нефункціональними перевірками. Центральне місце посідає функціональне тестування бізнес-логіки, де основним об'єктом є взаємодія між ролями Клієнта, Креатора та Продюсера. Стратегічний підхід передбачає глибоку верифікацію модуля Wallet, оскільки коректність нарахування коштів (де 1 G-pass дорівнює 1 Євро) та ідемпотентність транзакцій складають основу фінансової стабільності проєкту.

Особливу увагу у стратегії приділено тестуванню графічного інтерфейсу (GUI) та адаптивності верстки, що обумовлено технічною вимогою підтримки роздільних здатностей від 390px до 1920px. Стратегічний план включає кросбраузерні перевірки та симуляцію сенсорного керування для мобільних пристроїв, що забезпечує ідентичний користувацький досвід незалежно від типу термінала. Нефункціональний аспект стратегії також охоплює тестування безпеки та приватності, а саме валідацію механізму «Hide My Media» Креатора та перевірку ефективності автоматичної фільтрації текстового контенту за списками Banned Words.

Стратегія тестування GoSeeChat базується на ітераційному підході з чітким розмежуванням середовищ розробки. Первинна верифікація нових функцій проходить на Dev-сервері, що дозволяє локалізувати логічні помилки на ранніх етапах, тоді як на Prod-сервері виконується фінальне приймальне тестування (Smoke testing) для підтвердження цілісності системи перед релізом. Такий підхід дозволяє ефективно керувати ризиками регресії, коли впровадження нових алгоритмів (наприклад, системи Rockstar для пріоритезації Креаторів) може вплинути на стабільність існуючих модулів.

Процес управління дефектами, закладений у стратегію, передбачає використання інструментів діагностики, таких як Swagger UI для аналізу API-відповідей бекенду та Chrome DevTools для виявлення помилок у фронтенд-частині платформи. Це надає інженеру з якості можливість передавати розробникам вичерпну технічну інформацію для прискорення усунення дефектів.

Для забезпечення простежуваності процесу тестування та гарантування повного покриття технічного завдання використовується Матриця покриття вимог (Requirements Traceability Matrix — RTM). Цей інженерний інструмент встановлює прямий логічний зв'язок між бізнес-вимогами, зафіксованими у специфікації BRS, та розробленими тестовими сценаріями. Використання RTM дозволяє уникнути ситуацій, коли критичні функціональні зони, такі як ліміти завантаження медіафайлів (8 файлів для стандартного акаунта Креатора) або правила валідації нікнеймів Клієнтів (5–24 символи), залишаються неперевіреними.

Структурно матриця складається з ідентифікаторів вимог, їхнього короткого опису, посилань на відповідні тест-кейси та статусів їхнього виконання. У межах проєкту GoSeeChat матриця ведеться у хмарному середовищі Google Workspace, що забезпечує спільний доступ команди до актуальних даних про стан готовності продукту. Це дозволяє в реальному часі відстежувати відсоток успішно пройдених тестів за кожною роллю та модулем, що виступає ключовим індикатором для прийняття рішення про запуск системи у промислову експлуатацію.

Структуру матриці покриття вимог, яку інженер застосовує для верифікації ключових модулів платформи GoSeeChat, наведено в таблиці 2.2.

Розроблена стратегія тестування та матриця покриття вимог створюють цілісну методологічну базу для забезпечення високої якості соціальної платформи. Поєднання функціональних перевірок, аналізу адаптивності інтерфейсу та суворого контролю відповідності технічному завданню через інструмент RTM дозволяє мінімізувати технічні й фінансові ризики.

Таблиця 2.2 — Фрагмент Матриці покриття вимог (RTM) для критичних модулів GoSeeChat

ID Вимоги (BRS)	Опис технічної вимоги платформи	Пріоритет	ID Тест-кейсу	Статус покриття вимоги
BRS-REG-01	Реєстрація користувача із вибором базових тем інтересів (Topics)	High	TC-REG-001, TC-REG-002	Покрито (Passed)
BRS-FIN-04	Операції поповнення та списання внутрішньої валюти G-pass (1:1 до Євро)	High	TC-WAL-012, TC-WAL-015	Покрито (Passed)
BRS-MED-02	Ліміти завантаження медіафайлів Креатора у сховище Mediabank	Medium	TC-MED-044, TC-MED-045	Покрито (Passed)
BRS-PRIV-07	Активація функції приховування контенту Креатора «Hide My Media» від Клієнтів	High	TC-SEC-089, TC-SEC-090	Покрито (Passed)

Реалізація цієї стратегії гарантує загальну надійність системи, стабільність рольової взаємодії Клієнта, Креатора й Продюсера, а також безпеку обробки транзакцій у промисловому середовищі експлуатації.

2.4 Проєктування тестових сценаріїв та техніки тест-дизайну для обраних ролей.

Проєктування тестових сценаріїв (Test Case Design) у цій роботі базується на декомпозиції технічної специфікації BRS на атомарні функціональні вимоги.

Кожну з цих вимог інженер верифікує через призму конкретної ролі користувача. Задля забезпечення високої надійності системи, зокрема в частині фінансових модулів та захисту персональних даних, застосовано комбінований підхід. Він поєднує методи «чорної скриньки» (Black Box Testing) з аналізом станів архітектури додатка.

Технічні ліміти та відповідні тестові значення для проведення верифікації наведено в таблиці 2.3.

Таблиця 2.3 — Матриця граничних значень для ключових полів введення даних

Об'єкт верифікації	Технічний ліміт (BRS)	Тестові значення (BVA)	Очікуваний результат валідації системи
Нікнейм Клієнта	Від 5 до 24 символів	4, 5, 24, 25 символів	4, 25 — помилка (400 Bad Request); 5, 24 — успішно
Автентифікаційний пароль	Мінімум 8 символів	7, 8 символів	7 — відхилення запиту; 8 — успішна реєстрація
Поповнення балансу гаманця	Від 20 до 1000 G-pass	19, 20, 1000, 1001 G-pass	19, 1001 — блокування транзакції; 20, 1000 — успішно
Довжина текстового повідомлення	Межа тарифікації: 280 симв. Максимум: 4000 симв.	280, 281, 4000, 4001 символів	280 — стандартний тариф; 281 — преміум-блок; 4001 — блокування введення

При проектуванні сценаріїв для об'єктів із числовими та текстовими обмеженнями критичне значення має суміщення технік еквівалентного поділу (Equivalence Partitioning — EP) та аналізу граничних значень (Boundary Value Analysis — BVA). В інженерній QA-практиці це дозволяє локалізувати помилки

типу «off-by-one» (зсув на одиницю), які виникають у логіці валідації на рівні API та фронтенду платформи GoSeeChat.

Особливу увагу приділено логіці тарифікації текстових повідомлень. Межа у 280 символів є точкою переходу: повідомлення довжиною рівно 280 символів система тарифікує як стандартне (воно фіксується у діалоговому вікні протягом 60 секунд), тоді як 281 символ ініціює створення преміального платного блока. Тестові сценарії верифікують не лише списання валюти G-pass, а й коректність відображення елемента «Read more» у стрічці новин Клієнта, який має чітко підлаштовуватися під контури інтерфейсу без розриву рядка при перенесенні.

Для динамічних компонентів зі складною логікою переходів застосовано техніку тестування переходів станів (State Transition Testing). Сценарії проєктуються таким чином, щоб перевірити як валідні транзиції, так і неможливість переходу об'єкта в недопустимі стани під впливом некоректних дій користувача.

- життєвий цикл модуля збору коштів (Tip Box): сценарії охоплюють стани Draft, In Queue, Active, Archived та Banned. Інженер верифікує логіку автоматичної активації за настанням параметра Publication Date, а також автоматичне переведення в архів при досягненні 100% цільової суми, якщо активовано прапор «Unpublish if goal reached»;

- статуси анкет верифікації особистості (KYC): сценарії перевіряють переходи між станами Verification In Progress (первинне завантаження документів Креатором), Approved, Rejected (з описом причин та допуском до повторної подачі) та Blocked (безповоротне блокування бізнес-профілю).

Сценарії для ролі Клієнта зосереджені на забезпеченні безперервного користувацького досвіду та фінансовій безпеці операцій:

- багатфакторна реєстрація (OTP): верифікація унікальності одноразового коду, обмеження терміну його дії (15 хвилин) та ліміту спроб введення (не більше 5 спроб на 24 години) для запобігання брутфорс-атакам на рівні API;

- управління балансом Wallet: окрім базового поповнення (від 20 G-pass), сценарії включають обробку помилок при дефіциті коштів під час активного чату з Креатором. Система має миттєво викликати модальне вікно поповнення, призупиняючи тарифікацію сесії;

- контроль приватності «Hide My Media»: верифікація на рівні бекенду того, що при активації функції Креатором медіафайли стають заблюреними для Клієнтів, які не входять до його контакт-листа, а прямі URL-посилання на оригінальний контент повертають код помилки 403 Forbidden.

Для ролі Креатора сценарії фокусуються на управлінні інтелектуальною власністю та механізмах монетизації:

- верифікація через відеоідентифікацію: перевірка валідації текстового вмісту, який Креатор вимовляє на відеозаписі: «I, [Повне ім'я], content creator, confirm...». Тестовий сценарій контролює реакцію системи на завантаження некоректного формату файлу або відео без звукового потоку;

- робота підсистеми розсилок (Newsletter Engine): проєктування перевірок для трьох динамічних сегментів аудиторії: COLD (ознайомлення через базові теми інтересів), WARM (активні підписники профілю) та Follow-up. Сценарій верифікує обов'язкову стоп-умову: будь-яка платна дія з боку Клієнта має негайно виключити його з розсилки контуру COLD і перевести в категорію прямих контактів;

- логіка Mediabank (метод виключення): сценарії перевіряють зміну графічних маркерів файлів. Завантажений контент (первинна синя рамка) після призначення категорії (наприклад, Posts) отримує сіру рамку і автоматично зникає з переліку невідсортованих об'єктів.

Графічну модель зміни статусів контенту в межах бази даних Mediabank наведено на рисунку 2.4.



Рисунок 2.4 — Діаграма станів контенту в межах модуля Mediabank

Сценарії для ролі Продюсера мають найвищий рівень критичності через пряме управління фінансовими потоками підпорядкованих Креаторів:

- валідація правила «100% на рядок»: у фінансовій таблиці розподілу доходу сума часток Продюсера та підключених Креаторів має завжди дорівнювати 100%. Сценарій верифікує, що Креатор виступає в ролі балансуючого буфера: при збільшенні відсотка Продюсера система автоматично зменшує частку Креатора, але блокує операцію, якщо це значення намагаються зробити меншим за нуль;

- ієрархічна прив'язка профілів: верифікація автоматичного закріплення Креатора за конкретним Продюсером при ініціалізації інвайтингу, що виключає подвійне підпорядкування в межах однієї фінансової мережі;

- статистика доходів (Me / My Team): перевірка фінансових розрахунків загального доходу Продюсера та часток Креаторів з округленням результату до двох знаків після коми у більшу сторону для запобігання втратам на дрібних транзакціях;

Забезпечення безпеки інформаційного простору платформи реалізується через сценарії верифікації системи обмежень та автоматичної фільтрації:

- накопичення штрафних балів Креатором: моделювання ситуації, за якої кожна одиниця некоректно маркованого контенту категорії «18+» або «Sensitive» нараховує 1 штрафний бал. При досягненні 5 балів профіль Креатора отримує

статус «На обліку» (жовтий індикатор в інтерфейсі Продюсера), а при накопиченні 10 балів — бекенд автоматично блокує акаунт;

- валідація форми Help Center: перевірка обробки офіційних запитів, де система повинна валідувати Email заявника за списком дозволених державних доменів (.gov.ua, .gov.uk, .parliament.uk);

- перевірка на унікальність текстових блоків (Check text for duplicates): верифікація алгоритму порівняння полів «About Me» та «My Motto» нового Креатора з іншими профілями в базі даних для запобігання поширенню спаму.

Враховуючи специфіку напрямку комп'ютерної інженерії, розроблено технічні сценарії перевірки трансформації графічного інтерфейсу платформи:

- контур мобільного інтерфейсу (390px): перевірка появи нижнього елемента Bottom Navigation Bar з п'ятьма функціональними кнопками та трансформація аватара користувача у квадратну форму з закругленими кутами;

- контур десктопного інтерфейсу (1920px): верифікація статичного положення блока «My Messages» ліворуч від основного контенту та коректність роботи скролл-барів у модулі Mediabank;

- обрізка медіафайлів (Crop Media): перевірка функціонування модального вікна, де рамка обрізки зображення має центруватися незалежно від пропорцій завантаженого Креатором фото, а масштабування здійснюється через повзунок (слайдер) або сенсорні жести.

Розроблений комплекс сценаріїв забезпечує повне покриття функціональних та нефункціональних вимог платформи GoSeeChat, що є передумовою для випуску стабільного програмного продукту.

З ПРАКТИЧНЕ ВПРОВАДЖЕННЯ СИСТЕМИ ТЕСТУВАННЯ

Практичне впровадження системи тестування для соціальної платформи GoSeeChat завершує інженерний цикл забезпечення якості, переводячи теоретичні

моделі та спроектовані сценарії у площину верифікації в реальних умовах розробки програмного продукту. Цей етап передбачає перехід від абстрактного планування до безпосередньої перевірки функціональних модулів системи на основі аналізу технічних специфікацій BRS. Головна мета цього розділу полягає в демонстрації ефективності впровадженої стратегії ручного тестування, оцінці стабільності клієнт-серверної архітектури та перевірці надійності фінансових механізмів платформи.

Процес практичної апробації базується на декомпозиції системи за ролями користувачів, що забезпечує повне покриття технічних вимог для кожної групи учасників екосистеми: Продюсерів, Креаторів та Клієнтів. Такий підхід гарантує, що специфічна бізнес-логіка функціонує без відхилень від заданих параметрів. До ключових зон верифікації належать:

- алгоритми автоматичного нарахування та розподілу доходів у звітних таблицях Продюсера за правилом «100% на рядок»;
- механізми динамічної тарифікації текстових і відеоповідомлень у чатах Клієнта з урахуванням ліміту у 280 символів;
- процедури KYC-верифікації Креаторів із завантаженням ідентифікаційних документів та аналізом дублікатів анкет у базі даних.

Впровадження системи тестування на цьому рівні дозволяє валідувати не лише окремі інтерфейсні елементи, а й наскрізні сценарії взаємодії (End-to-End Testing), що безпосередньо формують користувацький досвід та забезпечують комерційну життєздатність програмного комплексу.

Організація процесу верифікації в умовах розподілених контурів підтримує безпечну локалізацію дефектів на ранніх етапах розробки. Використання Dev-сервера як основного майданчика для первинного тестування дозволяє інженеру з якості ідентифікувати логічні помилки та аномалії в обробці даних до моменту їхнього потрапляння у відкритий доступ.

Приймальне тестування на Prod-сервері виступає фінальним етапом контролю. Під час його виконання інженер підтверджує цілісність реляційної бази даних, валідність SSL-сертифікатів, а також коректність синхронізації між

фронтом на React та бекенд-сервером у реальному часі за умов використання протоколу WebSocket.

Обраний інструментальний комплекс пройшов повну апробацію під час виконання локальних тестових перевірок. Інженер застосовував Swagger UI для низькорівневої валідації JSON-відповідей сервера, аналізу кодів станів HTTP (400 Bad Request, 403 Forbidden) та підміни авторизаційних токенів Bearer Token для імітації сесій Клієнта або Продюсера.

Паралельно, за допомогою панелей Network та Application в інструментах Chrome DevTools, виконувався аналіз мережевої активності, фіксація затримок та моніторинг стану клієнтських сховищ Local Storage. Це дозволило підтвердити математичну точність фінансових транзакцій у внутрішній валюті G-pass, де кожен запис в історії гаманця (Wallet) суворо відповідає фактично здійсненій дії користувача. Такий підхід мінімізує ризики виникнення транзакційного дисбалансу та втрати цифрових активів.

Завершальним елементом практичної реалізації є систематизація виявлених невідповідностей через професійні системи управління дефектами. Аналіз метрик знайдених багів, класифікація їх за рівнем критичності (Blocker, Critical, Major, Minor) та проведення регресійного тестування дозволяють оцінити динаміку поліпшення стабільності продукту.

Зменшення щільності дефектів після повторних ітерацій перевірок підтверджує належну якість коду. У підсумку, практична реалізація не просто документує процес виконання чек-лістів, а формує об'єктивну інженерну оцінку готовності платформи GoSeeChat до запуску в промислову експлуатацію.

3.1 Тестування функціональних модулів за ролями

Практична стадія верифікації програмного комплексу GoSeeChat базується на переході від загальних стратегій до цільового тестування специфічних

функціональних вузлів, що формують бізнес-логіку системи для кожної окремої ролі користувача. Цей процес реалізується через виконання структурованого документа «Checklist GoSeeChat», який виступає результатом декомпозиції технічної специфікації бізнес-вимог (BRS) на набір конкретних перевірок.

Кожна роль у системі — Клієнт, Креатор та Продюсер — розглядається як окремий об'єкт дослідження з унікальними правами доступу, лімітами та сценаріями взаємодії з бекенд-інфраструктурою платформи. Такий підхід забезпечує ізольовану перевірку модулів та мінімізує ризики виникнення дефектів на стику прав різних категорій користувачів.

Основним інструментом практичного виконання перевірок є «Checklist GoSeeChat», що містить деталізовані графи:

- назва бізнес-ролі;
- найменування функціональної секції;
- опис перевірки;
- передумови (Preconditions);
- очікуваний результат (Expected Result);
- кількісна оцінка трудомісткості (QA estimate).

Використання цього інструмента дозволяє системно охопити всі рівні архітектури, починаючи від автентифікації через OAuth-протоколи (Google, Apple ID, Facebook) і завершуючи складними фінансовими операціями у внутрішній валюті G-pass. У ході тестування інженер приділяє особливу увагу валідації вхідних даних, де кожне поле форми введення — від нікнейма користувача до платіжних реквізитів — проходить перевірку на відповідність граничним значенням та правилам безпеки, які встановлює специфікація BRS.

Технічна діагностика модулів під час проходження чек-ліста здійснюється за допомогою засобів розробника Chrome DevTools та інтерфейсу Swagger UI. Swagger UI використовується як основний засіб верифікації ідемпотентності транзакцій та цілісності JSON-відповідей сервера, що критично для модуля Wallet.

Наприклад, при тестуванні поповнення балансу (Top Up) або списання коштів за платні повідомлення довжиною понад 280 символів, тестувальник валідує не лише візуальне оновлення інтерфейсу, а й коректність статусів транзакцій та наявність обов'язкового булевого поля isPaid у відповіді API. Це гарантує математичну точність фінансових розрахунків, де 1 одиниця G-pass суворо еквівалентна номіналу в 1 Євро.

Верифікація графічного інтерфейсу (GUI) у межах цього розділу зосереджена виключно на десктопній версії платформи з фіксованою роздільною здатністю 1920x1080 px. Відповідно до поточного стану розробки проєкту, адаптивність мобільних та планшетних версій інтерфейсу (роздільні здатності 390 px та 768 px) на цьому етапі виключена з обсягу робіт. Ці модулі перебувають у стадії активного формування дизайну та логіки перемикання елементів мобільної навігації.

Тому тестування верстки обмежується перевіркою коректності відображення статичних елементів хедера, сайдбара з повідомленнями та модальних вікон, де контролюється центрування рамки обрізки контенту та працездатність слайдерів масштабування на великих екранах.

Функціональне тестування ролі Клієнта охоплює повний цикл споживання контенту: від вибору тематичних інтересів (Topics) під час онбордингу до використання системи розсилок Newsletter. Ключовим об'єктом верифікації тут є тарифікація чатів, де система має коректно об'єднувати повідомлення в один часовий блок протягом 60 секунд або ініціювати преміальне списання за контент великого обсягу.

Для ролі Креатора фокус зміщується на управління інтелектуальною власністю в модулі Mediabank, де тестується логіка «методу виключення» при сортуванні файлів за категоріями та коректність відображення кольірних індикаторів статусів медіа (Active, In Queue, Banned).

Верифікація ролі Продюсера є найбільш технічно складною, оскільки вона включає управління ієрархією підключеної команди Креаторів та контроль таблиці розподілу доходу, де діє суворе інженерне правило «100% на рядок».

У наступних підпунктах наведено детальний опис результатів проходження тестових сценаріїв для кожної ролі, що дозволяє сформувати цілісну картину якості платформи GoSeeChat та підтвердити її готовність до промислової експлуатації на рівні десктопного вебрішення. Кожна перевірка, зафіксована у практичній частині, супроводжується аналізом відповідності фактичної поведінки системи вимогам BRS, що забезпечує наукову обґрунтованість висновків щодо стабільності клієнт-серверної взаємодії.

3.1.1 Тестування функцій ролі Продюсера

У межах архітектурної моделі платформи GoSeeChat роль Продюсера функціонує як високорівневий аналітичний та адміністративний центр. Цей інтерфейс інтегрує інструменти управління персоналом, контролю фінансових транзакцій, модерації контенту та крос-рольової координації.

З погляду комп'ютерної інженерії, верифікація цього модуля вимагає комплексного підходу: від перевірки статичних елементів графічного інтерфейсу (GUI) до дослідження складних логічних зв'язків між реляційними базами даних та алгоритмами розподілу фінансових коштів.

Процес тестування ролі Продюсера орієнтований на десктопну версію платформи з фіксованою роздільною здатністю 1920x1080 рх. Таке обмеження обумовлене специфікою бізнес-функцій, які передбачають оперування великими масивами табличних даних, що потребують стабільної фіксації хедера та підтримки безпомилкової роботи горизонтального скролу.

При цьому, відповідно до вказаних обмежень предметної області, верифікація всіх наскрізних сценаріїв та ієрархічних зв'язків у межах ролі Продюсера (включаючи таблиці розподілу доходів, призначення підпорядкованих профілів, масові операції та фінансову статистику) здійснюється із фокусом виключно на трьох базових суб'єктах платформи: Продюсері, Креаторів та Клієнтах.

Первинна стадія апробації системи тестування охоплює перевірку компонента «Producer Registration», де реалізовано механізм поетапного збору та валідації персональних даних. На першому кроці інженер перевіряє роботу алгоритмів валідації полів введення на рівні фронтенду та бекенду:

- username — суворе обмеження довжини в межах 5–24 символів;
- password — вимога щодо наявності мінімум 8 символів, включаючи літери верхнього й нижнього регістрів, цифрові знаки та спеціальні символи;
- email — валідація синтаксису адреси електронної пошти та перевірка наявності доменів, дозволених для офіційної комунікації і надсилання юридичних запитів.

Програмна логіка блокує кнопку «Continue» до моменту повного заповнення всіх обов'язкових полів атрибутів, що підтверджують повноліття користувача (на основі дати народження) та визначають країну його податкового резидентства.

На другому етапі реєстрації виконується верифікація особистості користувача. Тут тестуванню підлягає функціонал завантаження цифрових копій документів та ідентифікаційних фотознімків. В інженерному аспекті досліджується робота модального вікна Stop Media, яке забезпечує центрування та обрізку графічних об'єктів до заданих пропорцій.

Імітація завантаження файлів різних форматів (PNG, JPG, GIF) дозволяє підтвердити, що система коректно обробляє встановлені технічні ліміти: обсяг файлу не повинен перевищувати 15 Мб, а максимальна роздільна здатність становить 5000x5000 пікселів. Окремо перевіряється життєвий цикл станів анкети після відправлення запиту на сервер, де об'єкт послідовно змінює статуси за схемою: Approval (очікування перевірки) → Approved (акцептовано), Rejected (відхилено з описом причин) або Blocked (безповоротне блокування профілю).

Управлінська логіка ролі Продюсера базується на створенні та закріпленні зв'язків із Креаторами. Процес тестування інтерфейсу передбачає перевірку механізму динамічного зв'язування даних. Коли Продюсер ініціює зміни у структурі управління для конкретного Креатора, система повинна автоматично оновити реляційні зв'язки в базі даних та відобразити актуальну ієрархію в таблицях «Creators». Верифікація цієї логіки здійснюється шляхом маніпуляцій з ідентифікаторами URL-запитів та моніторингу результируючих об'єктів.

В межах цього підрозділу також аналізуються масові дії (Bulk Actions). Продюсер має можливість виділити групу Креаторів та через єдине модальне вікно змінити параметри їхнього обслуговування або статус доступу (Full/Limited). Інженер з якості валідує такі умови:

- елементи керування масовими операціями активуються лише за умови вибору сумісних записів;
- при спробі переведення Креатора в статус Published без наявності в базі даних обов'язкового мінімуму контенту (5 фотографій, 5 відеофайлів та 1 публікації), система генерує блокуюче модальне попередження з кодом помилки валідації;
- функція Unlink Team миттєво розриває поточні адміністративні зв'язки з обраним Креатором, проте повністю зберігає його історичну фінансову статистику для уникнення втрати даних.

Модуль «Content Moderation» виступає центральним вузлом контролю якості публікацій. Тестування охоплює перевірку обробників подій для елементів Accept та Reject. При відхиленні публікації інженер перевіряє обов'язковість вибору причини (наприклад, Adult content, Fraud, Scams) та коректність фіксації порушень. Накопичення критичної кількості скарг (Reports) від Клієнтів автоматично змінює колірний індикатор анкети Креатора на червоний та блокує доступ до платних функцій взаємодії.

Окремим об'єктом тестування є функція Check text for duplicates, інтегрована в текстові поля «About Me» та «My Motto». Тестувальник верифікує роботу алгоритму порівняння текстових рядків. Система повинна аналізувати базу даних і підсвічувати збіги з іншими профілями для запобігання поширенню автоматизованого спаму. Цей механізм перевіряється шляхом введення ідентичних текстів для різних акаунтів та фіксації реакції бекенду через Swagger UI, де у тілі JSON-відповіді сервера має з'явитися прапор "isDuplicate": true.

Найбільш критичним з точки зору фінансової безпеки є тестування таблиці розподілу доходів та інтерфейсу «Compensation». Платформа оперує внутрішньою валютою G-pass (де 1 G-pass дорівнює 1 Євро). Суворій перевірці підлягає

інженерне «правило 100% на рядок»: Продюсер встановлює фіксовану відсоткову частку для себе, а частка Креатора розраховується автоматично як залишок від 100%. Інженер з якості перевіряє граничні значення через границі BVA: система блокує спроби встановити відсоток Продюсера понад 100%, а частка Креатора при цьому програмно захищена від переходу в область від’ємних значень.

На сторінці «Balance» верифікується життєвий цикл виплат (Payouts). Тестуванню підлягають статуси транзакцій:

- in Process (транзакцію ініційовано та зафіксовано в системі);
- approved (запит схвалено фінансовим контуром платформи);
- paid out (кошти успішно перераховано, інтерфейс змінює маркер на зелений колір);
- declined (у виплаті відмовлено, а заблокована сума автоматично повертається на баланс гаманця Продюсера).

Під час перевірки компонента «Bank Details» тестувальник валідує поля міжнародних форматів IBAN, SWIFT та працездатність функції Set as default для вибору основного розрахункового рахунку.

Фрагмент практичного чек-ліста, що використовувався під час інструментальної верифікації ролі Продюсера, наведено в таблиці 3.1 [9].

Таблиця 3.1 — Фрагмент структурованого чек-ліста для інтерфейсу Producer

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
PR-REG-01	Producer Registration	Валідація довжини поля Username на граничних значеннях (4, 5, 24, 25 симв.)	Відкрита форма першого кроку реєстрації Продюсера	4, 25 симв. — блокування; 5, 24 симв. — запит успішний	5 хв

Продовження таблиці 3.1

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
PR-KYC-04	KYC Verification	Завантаження файлу Selfie with ID обсягом 151 Мб у форматі PNG	Анкета у статусі Approval, активовано вікно Story Media	Система видає помилку валідації розміру, файл не завантажується	5 хв
PR-FIN-02	Compensation Model	Перевірка «правила 100% на рядок» при введенні частки Продюсера у 102%	Відкрита таблиця розподілу часток доходу в ролі	Блокування збереження даних, виведення попередження про перевищення ліміту	5 хв
PR-UI-07	Producer finstats	Перевірка стабільності фіксації хедера таблиці при вертикальному скролі	Завантажена сторінка статистики, наявні >30 записів Креаторів	Хедер залишається фіксованим у верхній точці, дані зміщуються під нього	5 хв

Оскільки роль Продюсера спроектовано як складний аналітичний інструмент, тестування графічного інтерфейсу зосереджено на стабільності відображення елементів при базовій роздільній здатності 1920x1080 px. Інженер з якості перевіряє, що лівий навігаційний сайдбар зберігає статичне положення, а

центральна область (Newsfeed або Fin Stats) коректно підтримує пагінацію із фіксованим вибором 30, 60 або 90 записів на одну сторінку інтерфейсу.

Кожен інтерактивний елемент, включаючи графічні іконки редагування профілю чи кнопки фільтрації даних за календарем, перевіряється на наявність візуального ефекту наведення курсору (hover) та точність редиректів на відповідні URL-адреси підрозділів додатка. При тривалому вертикальному скролі вниз інтерфейс повинен динамічно відображати кнопку швидкого повернення вгору (елемент «↑»), що спрощує навігацію у великих списках підключених Креаторів. Technical-верифікація через консоль інструментів Chrome DevTools підтверджує повну відсутність JavaScript-помилки (Uncaught Unhandled Exceptions) під час ініціалізації модальних вікон Account Status History або Duplicate Check.

3.1.2 Тестування функцій ролі Креатора

У межах архітектурної моделі платформи GoSeeChat роль Креатора є центральним елементом екосистеми, оскільки цей профіль забезпечує генерацію, систематизацію та безпосередню монетизацію контенту, який споживають Клієнти. Практичне тестування інтерфейсу Креатора охоплює комплекс заходів з верифікації інструментів персоналізації профілю, управління інтелектуальною власністю в модулі Mediabank, а також аналізу алгоритмів нелінійної взаємодії з аудиторією через підсистему розсилок Newsletter Engine.

Зважаючи на інженерну специфіку комп'ютерної інженерії, усі тестові сценарії виконувалися на десктопній версії вебдодатка з базовою роздільною здатністю 1920x1080 px. Це дозволило детально проаналізувати поведінку складних елементів графічного інтерфейсу (GUI), таких як динамічні лічильники, інтерактивні редактори публікацій та багаторівневі форми введення даних [10].

Відповідно до встановлених меж предметної області, верифікація бізнес-сценаріїв ролі Креатора фокусується виключно на його взаємодії з Клієнтами (споживачами послуг) та Продюсером (координатором фінансових потоків).

Практична верифікація розпочалася з тестування сторінки «General», яка виконує функцію ядра ідентифікації Креатора. Технічна специфікація BRS

визначає масив обов'язкових полів для успішної ініціалізації акаунта: Intro photo, Cover photo, Name, Nickname, Description, Response time, Topics та My Motto.

Під час проходження структурованого чек-ліста інженер перевіряв логіку валідації текстових блоків «About Me» та «My Motto». Алгоритми обробки рядків на рівні бекенду мають автоматично видаляти зайві пробіли (процедура trimming) на початку та в кінці введеного значення, а також блокувати спроби ін'єкцій програмного коду (HTML-теги, JavaScript-сценарії, inline-елементи iframe), повертаючи системне повідомлення про помилку: «Invalid content: links and code are not allowed» [11].

Окремо верифіковано параметр «Response time» (SLA-ліміт відповіді на запити Клієнтів), що варіюється в межах від 1 до 7 днів. Тестування підтвердило, що цей показник безпомилково записується в базу даних та інтегрується в алгоритм «Rockstar», який ранжує та пріоритезує профілі Креаторів у глобальній системі пошуку.

Особливу увагу приділено інженерному аналізу модуля Mediabank, який функціонує за принципом централізованого об'єктного сховища файлів Креатора. Процес тестування базувався на валідації логіки «методу виключення» та системи колірних індикаторів, що відображають поточний стан життєвого циклу медіаконтенту. Інженер з якості перевіряв коректність роботи динамічних лічильників інтерфейсу, де колірні маркери розподіляються таким чином:

- сірий колір рамок/лічильника — нові, нерозподілені файли, завантажені з пристрою;
- зелений колір — активні публікації, доступні Клієнтам у стрічці новин;
- жовтий колір — запланований контент, що перебуває у черзі публікації;
- червоний колір — матеріали, заблоковані за результатами модерації Продюсера.

Верифікація підтвердила, що після призначення файлу до певної категорії (наприклад, переведення у розділ «Posts»), об'єкт отримує сірий маркер і автоматично виключається з переліку доступних для інших розділів. Це запобігає дублюванню контенту в базі даних. Технічне тестування також охопило валідацію

лімітів завантаження: система стабільно обробляє файли форматів GIF, JPEG, PNG обсягом до 15 Мб, блокуючи транзиції при перевищенні роздільної здатності 5000x5000 пікселів на рівні API-відповідей (код 400 Bad Request) [12].

Модуль створення та редагування публікацій (Posts та Vids) пройшов серію сценаріїв, що моделюють реальну активність Креатора. При тестуванні текстових постів верифікувалася унікальна UX-вимога щодо кнопки «Read more»: система повинна автоматично приховувати кілька останніх слів перед кнопкою, якщо її поява припадає на початок нового рядка, що підтримує візуальну естетичність відображення довгих текстів (понад 280 символів) у стрічці Newsfeed Клієнта.

Для розділу «Vids» перевірено логіку виставлення дати відкладеної публікації. Тестування підтвердило, що спроба введення дати в минулому часі ініціює помилку валідації форми, а успішно заплановані відео отримують статус «In Queue» і автоматично публікуються згідно з серверним часом. Кожна публікація додатково перевірялася на коректність встановлення міток «18+» та «Sensitive», які мають ініціювати накладання ефекту розмиття на клієнтській стороні для неавторизованих користувачів або Клієнтів із відповідними обмеженнями.

Технічно складним етапом стала верифікація системи розсилок «Newsletter Engine», що вимагала перевірки бізнес-логіки сегментації аудиторії Клієнтів. Інженер з якості валідував коректність розподілу одержувачів за динамічними групами: COLD (нові Клієнти, залучені через вибір тем інтересів), WARM (активні шанувальники) та Follow-up (цільова група для повторної комунікації).

Під час виконання тестів перевірено працездатність динамічних тегів персоналізації, таких як {Name} та {Nickname}, які автоматично підставляють персональні дані Клієнта в тіло повідомлення розсилки обсягом до 4000 символів. Ключовим результатом стало підтвердження стабільності «стоп-умов»: система автоматично призупиняє заплановані розсилки, якщо Креатор перебуває у статусі офлайн понад 48 годин або накопичує понад 100 невідповідей у чатах, що супроводжується виведенням інформаційного банера в ролі користувача.

Завершальним етапом практичного впровадження став аналіз модулів фінансової та робочої статистики («Fin Stats» та «Work Stats»). Під час тестування підтверджено використання WebSocket-з'єднань для оновлення балансу в реальному часі: при здійсненні платної дії Клієнтом, дохід Креатора оновлюється миттєво без потреби перезавантаження сторінки додатка.

Тестувальник верифікував точність нарахування внутрішньої валюти G-pass (1 од. = 1 Євро), переконуючись, що система коректно вираховує і знімає фіксований відсоток Продюсера перед зарахуванням фінальної суми на баланс Креатора. Дані в таблицях підтримують горизонтальний скрол та фільтрацію за типами активностей (Tips, G-gift, Videochat), що дозволяє Креатору аналізувати структуру свого прибутку [13].

Додатково було верифіковано коректність обробки нестабільних мережових з'єднань під час тривалих сесій Креатора. У разі імітації втрати пакетів або переходу в офлайн-режим, інтерфейс миттєво блокував інтерактивні форми введення даних і відображав повідомлення про помилку зв'язку, запобігаючи втраті незбереженого контенту та некоректній відправці запитів на бекенд.

Фрагмент практичного чек-ліста, застосованого під час верифікації ролі Креатора, наведено в таблиці 3.2.

Таблиця 3.2 — Фрагмент «Checklist GoSeeChat» для ролі Creator

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CR-GEN-01	General (Профіль Креатора)	Додавання та збереження первинного медіафайлу Intro photo.	Користувач авторизований у бізнес-ролі Креатора, відкрита форма «General».	Об'єкт завантажується, відображається інтерфейсі, активується елемент Edit.	2 хв

Продовження талиці 3.2

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CR-GEN-02	General (Профіль Креатора)	Валідація текстового блоку «About Me» на наявність inline-скриптів та HTML-тегів.	Форма редагування профілю відкрита, в полі опису вноситься рядок: <code><script>alert('test')</script></code> .	Перехоплення запиту, блокування відправки, виведення помилки: «Invalid content: and code are not allowed».	3 хв
CR-MED-01	Mediabank (Сховище медіа)	Моніторинг та архівація об'єктів через сірий лічильник медіафайлів.	Локальні файли завантажені в систему, але не прив'язані до функціональних категорій платформи.	Показник сірого лічильника інтерфейсу відповідає масиву нових несортованих об'єктів.	2 хв
CR-MED-02	Mediabank (Сховище медіа)	Тестування логіки «методу виключення» категоризації завантаженого контенту.	Об'єкт зі статусом New (синя рамка) переводиться інженером у категорію публікацій «Posts».	Маркер файлу на сірий колір, виключається з переліків інших розділів.	3 хв

Продовження талиці 3.2

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CR-PST-01	Posts (Створення публікацій)	Перевірка унікального UX-алгоритму відображення та перенесення елемента «Read more».	Створюється пост із текстовим блоком довжиною понад 280 символів у стрічці Newsfeed.	Якщо межа переходу тексту збігається з початком рядка, система автоматично приховує крайні слова, запобігаючи візуальному розриву верстки.	3 хв
CR-NWL-01	Newsletter (Система розсилок)	Автоматична ідентифікація забороненого контенту (Banned Words) у Newsletter Engine.	Контур створення масової розсилки, введення тексту, що містить стоп-слова згідно зі специфікацією безпеки.	Інтерфейс підсвічує заборонені червоним кольором і блокує виконання розсилки.	4 хв
CR-VID-01	Vids (Короткі відеоролики)	Валідація часових параметрів при плануванні відкладеної публікації відеоконтенту.	Модуль «Vids», активовано календар планування розкладу, ініціюється вибір часової мітки у минулому часі.	Форма введення блокується, помилка валідації дати, надсилання невалідного POST-запиту на сервер унеможливлено	2 хв

Продовження талиці 3.2

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CR-FIN-01	Fin Stats (Фінансова статистика)	Валідація точності нарахування G-pass через протокол WebSocket при отриманні оплати від Клієнта.	Клієнт виконує платну дію (читання повідомлення >280 симв.), відкрита сторінка статистики Креатора.	Баланс гаманця оновлюється в реальному часі без перезавантаження сторінки, сума зараховується з вирахуванням фіксованого відсотка Продюсера.	5 хв

У підсумку, комплексне ручне тестування ролі Креатора підтвердило належну надійність інструментів генерації та монетизації контенту на платформі GoSeeChat. Виконання чек-ліста дозволило виявити та усунути дефекти в логіці обробки дублікатів анкет та налаштуваннях тригерів розсилок, що гарантує стабільну роботу системи в умовах інтенсивної клієнт-серверної взаємодії.

3.1.3 Тестування функцій ролі Клієнт

Верифікація клієнтського досвіду (Customer Experience) є фінальним етапом тестування функціональних модулів за бізнес-ролями, де перевірка підлягає повний життєвий цикл користувача — від первинної автентифікації до виконання преміальних фінансових транзакцій. Тестування ролі Клієнт у межах комп'ютерної інженерії розглядається як валідація складної клієнтської логіки, яка забезпечує взаємодію з бекенд-сервером через архітектурні підходи REST API та WebSocket-з'єднання для реалізації функцій обміну даними в реальному часі.

Основний акцент під час випробувань зміщується на перевірку цілісності фінансових потоків у підсистемі Wallet, коректність тарифікації мультимедійного

контенту та ергономіку графічного інтерфейсу на десктопній роздільній здатності 1920x1080 px.

Відповідно до визначених меж предметної області, дослідження всіх наскрізних клієнтських сценаріїв (включаючи обмін повідомленнями, тарифікацію дзвінків та купівлю контенту) проводиться із залученням виключно трьох базових суб'єктів платформи: Клієнта, Креатора та Продюсера.

Верифікація механізмів автентифікації та онбордингу

Процес тестування розпочинається з модулів Log In та Sign In, де верифікується інтеграція декількох методів доступу: через поштову скриньку (Email) з відправкою одноразового пароля OTP та за допомогою безпекових протоколів OAuth 2.0 (Google, Apple ID, Facebook, Twitter). Особлива увага приділяється сценарію відновлення доступу (Forgot password), де тестувальник валідує безпекову вимогу специфікації: після 5 послідовних невдалих спроб генерації та відправки перевірного коду система повинна заблокувати цю функцію для конкретного ідентифікатора Email на 24 години, запобігаючи атакам типу Brute Force [14].

Критичним вузлом онбордингу виступає модальне вікно «Topics», яке ініціюється після первинної реєстрації акаунта Клієнта. Згідно з технічним завданням BRS, система надає п'ять груп інтересів, серед яких Auto/Sport, Fashion/Beauty, Art/Languages. Тестувальник перевіряє логіку станів керуючих елементів: керуюча кнопка «Thank you» залишається програмно заблокованою до моменту вибору користувачем хоча б одного чекбокса в масиві доступних тем. Це забезпечує подальше коректне релевантне формування персоналізованої стрічки контенту на основі вподобань Клієнта.

Центральною інженерною логікою клієнтської зони є механізм автоматичного перемикавання між вкладками «GoSee» та «Contacts» у блоці «My Messages». Інженер з якості верифікує зміну станів інтерфейсу на основі обчислюваних параметрів бази даних:

- сценарій 1: Якщо у Клієнта зафіксовано ≤ 3 активних двосторонніх діалогів із Креаторами, за замовчуванням відкривається вкладка «GoSee» для стимулювання пошуку нових профілів;

- сценарій 2: Якщо кількість активних чатів становить >3 , пріоритет відображення зміщується на вкладку «Contacts».

Верифікація проводиться з урахуванням того, що архівовані та заблоковані діалоги не враховуються алгоритмом при визначенні кількості активних чатів. Паралельно тестувальник перевіряє індикатори непрочитаних повідомлень (Unread Badges), які повинні динамічно оновлюватися в реальному часі через WebSocket-канал без затримок верстки.

Модулі глобального пошуку Креаторів, фото- та відеоматеріалів проходять верифікацію на предмет коректності фільтрації та відпрацювання механізмів пагінації даних. На сторінці «Creators» тестується логіка порційного завантаження карток профілів: одночасно підвантажується масив із 6 об'єктів, після чого ініціюється поява кнопки «Next» для переходу до наступної групи елементів. При тривалому вертикальному скролі вниз верифікується динамічна поява елемента навігації зі стрілкою «↑» для миттєвого повернення до початку стрічки.

Процес взаємодії з медіаконтентом Креаторів (Posts, Videos) включає перевірку таких інженерних умов:

- функція «Read more»: Елемент керування не повинен ініціюватися на початку нового рядка. Якщо довжина тексту перевищує ліміт, алгоритм обробки на фронтенді автоматично приховує крайні слова перед кнопкою, запобігаючи розриву верстки у стрічці Newsfeed;

- параметри приватності (Blur): Перевіряється автоматичне накладання ефекту розмиття зображення для фото- та відеофайлів, що мають мітки «18+» або «Sensitive», до моменту явного підтвердження згоди на перегляд з боку Клієнта;

- модуль закладок (Bookmarks): Валідуються умова, за якої додані публікації миттєво з'являються у відповідному розділі навігаційного меню.

Тестування системи Wallet базується на суворому дотриманні паритету внутрішньої валюти: 1 G-pass еквівалентний 1 Євро. Інженер з якості валідую

процес поповнення балансу (Top Up) через підключені шлюзи Verotel та PassimPay, контролюючи встановлені ліміти (мінімальний поріг становить 20 G-pass) та точність редиректів на цільову сторінку /payment-info.

В історії платежів (Payment history) верифікується відповідність маскуванню номерів платіжних карток за міжнародним стандартом (відображення у форматі **** 1234), а також точність логування транзакцій із прив'язкою до системних типів: «Video in», «Message 281+» або «G-gift».

Преміум-функціонал (Upgrade account) тестується для трьох рівнів привілеїв (Level 1, 2, 3). Перевіряється, що після успішного підтвердження транзакції ліміти на завантаження медіа в особисте сховище та гранична кількість створення нотаток (збільшення базового ліміту з 8 до 112 одиниць) оновлюються автоматично без перезавантаження вебсторінки.

Модуль двостороннього чату є найбільш динамічним вузлом, де верифікації підлягає складна логіка тарифікації дій Клієнта під час взаємодії з Креатором:

- текстові повідомлення: Текстові пакети довжиною до 280 символів включно тарифікуються за стандартною базовою ставкою чату (0.59 Євро згідно з BRS). Починаючи з 281 символу, система ініціює преміальне списання фіксованої суми (3.49 Євро) з балансу гаманця Клієнта;
- відеочат: Критичний сценарій, де бізнес-логіка передбачає запуск тарифікації сесії лише після 3 секунд стабільного з'єднання. Перевіряється можливість безкоштовного скасування виклику протягом перших 3 секунд, коли запит ще не передається на термінал Креатора;
- мультимедійні вкладення: Перевіряється відправка («відеокружечків»), аудіозаписів та графічних стікерів. Заблоковані для поточного рівня Клієнта стікери повинні відображатися з іконкою замка в золотому колі;
- шаблони нотаток (Notes): Верифікується вставка текстових шаблонів у чат. Якщо в структурі шаблону виявлено лексему з переліку стоп-слів, кнопка відправки блокується червоним підсвічуванням некоректного тексту.

Структуру технічних перевірок для ролі Клієнта, які виконувалися під час проходження практичного чек-ліста, наведено в таблиці 3.3.

Окрему увагу було приділено симуляції критичних станів мережі за допомогою інструментів розробника, вкладка Network Chrome Devtools, (Network Throttling) під час активних сесій Клієнта в текстовому та відеочатах. Тестування підтвердило, що при штучному обмеженні швидкості з'єднання до профілю Slow 3G, зниження швидкості мережі або короткочасному розриві зв'язку (Offline mode), клієнтська частина додатка коректно обробляє затримки відповідей від WebSocket-сервера.

Таблиця 3.3 — Фрагмент структурованого чек-ліста для бізнес-ролі Client

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CL-LOG-01	Log In (Авторизація)	Автентифікація Клієнта через сторонні сервіси Google/Apple ID.	Користувач не авторизований, відкрита форма входу на десктопній версії.	Успішне створення сесії, редирект в роль, дані профілю синхронізовані.	3 год
CL-ONB-01	Topics (Онбординг)	Валідація логіки вибору обов'язкових інтересів Клієнта на формі онбордингу.	Первинний вхід після реєстрації, ініційовано поп-ап вікно «Topics».	Кнопка «Thank you» активується після вибору мінімум однієї групи.	2 хв
CL-WLT-01	Wallet (Гаманець)	Перевірка ліміту мінімального поповнення балансу гаманця через шлюз Verotel.	Авторизований Клієнт, відкрита сторінка Top Up, введено суму \ge 20 G-pass.	Редирект на платіжну систему, транзакція отримує статус In Process.	5 хв

Продовження таблиці 3.3

ID	Функціональна секція	Опис тестової перевірки	Передумови виконання	Очікуваний результат (BRS)	QA Estimate
CL-WLT-02	Wallet (Гаманець)	Перевірка маскування конфіденційних даних банківської картки в історії платежів.	Здійснено транзакцію, відкрита вкладка Payment history.	Номер платіжного засобу відображається у захищеному форматі ****1234.	5 хв
CL-CHT-04	Chat (Обмін даними)	Валідація алгоритму зміни тарифікації текстового повідомлення довжиною 281 символ.	Відкрите вікно активного чату з Креатором, введено рядок обсягом 281 символ.	Система ініціює списання G-pass за преміальним тарифом для повідомлень типу Msg In 281+.	2 хв
CL-VCH-02	Video Chat (Відеодзвінки)	Перевірка роботи 3-секундного ліміту для безкоштовного скасування виклику.	Ініційовано вихідний відеодзвінок, запущено таймер сесії.	Кнопка Cancel активна перші 3 секунди, на 4-й секунді зникає, запускаючи тарифікацію.	3 хв
CL-PRM-01	Premium (Апгрейд)	Апгрейд рівня облікового запису користувача до преміального статусу (Level 3).	Баланс гаманця достатній, активовано елемент Upgrade account.	Ліміти створення нотаток в інтерфейсі збільшуються з 8 до 112 одиниць без релоаду сторінки.	10 хв

Продовження таблиці 3.3

CL-SET-03	Privacy (Безпека)	Перевірка працездатності безпекового тогла «Hide My Media» у налаштуваннях профілю.	Клієнт активує функцію обмеження видимості розділі Settings.	Усі медіафайли Клієнта автоматично блокуються для користувачів, яких немає в його Contacts.	5 хв
-----------	-------------------	---	--	---	------

Завершальний етап тестування включає перевірку компонентів розділу Settings. У межах графічного інтерфейсу верифікується коректність локалізації та зміни мови системи (доступні мовні пакети: Українська, Англійська, Німецька, Французька), інструменти управління груповими сповіщеннями, а також наскрізна логіка повного видалення облікового запису з деактивацією сесій. Особливе значення має перевірка працездатності функції безпеки «Hide My Media». При її активації бекенд-сервер повинен миттєво обмежити доступ до медіатеки Клієнта для всіх профілів, які не мають підтвердженого статусу в реляційній таблиці контактів (Contacts) поточного користувача.

Практичне ручне тестування клієнтського досвіду підтвердило стабільність основних бізнес-циклів вебплатформи GoSeeChat на рівні взаємодії Клієнта з Креаторами та Продюсером. Застосування технік аналізу граничних значень (BVA) та еквівалентного розділення для модулів Wallet та чат-сесій дозволило гарантувати точність розрахунків у внутрішній валюті G-pass та виключити ризики транзакційних помилок.

Попри виключення перевірок мобільної та планшетної адаптивності на поточному етапі життєвого циклу розробки програмного продукту, десктопне вебрішення продемонструвало повну відповідність вимогам технічного завдання UX/UI, здатність коректно ініціалізувати та обробляти фінансові операції, та високу надійність архітектурних механізмів захисту конфіденційних даних користувачів.

3.2. Процес тестування в умовах розподілених середовищ (Dev/Prod)

Організація процесу забезпечення якості для соціальної платформи GoSeeChat базується на фундаментальному принципі системної інженерії — суворому розмежуванні контурів розробки та промислової експлуатації. В умовах розгортання високонавантажених клієнт-серверних систем, що оперують фінансовими транзакціями та конфіденційними даними, використання розподілених середовищ є обов'язковою умовою для мінімізації технічних та репутаційних ризиків. Інфраструктура проєкту GoSeeChat функціонує через два ізольовані за рівнем доступу сервери: Dev-сервер (Development/Staging) та Prod-сервер (Production).

Dev-сервер виступає основним інженерним майданчиком для первинної верифікації функціональних модулів та виправлених дефектів програмного коду. Ця ізольована платформа дозволяє інженеру з ручного тестування проводити глибоку технічну діагностику системи, не впливаючи на стабільність роботи активних користувачів додатка. Основна мета цього середовища полягає в локалізації архітектурних та логічних помилок на ранніх ітераціях: грубих збоїв бізнес-логіки, дефектах валідації вхідних полів або некоректних структур відповідей серверної частини.

На Dev-сервері тестувальник оперує переважно тестовими (mock) даними, що забезпечує безпечну перевірку граничних значень. Наприклад, при верифікації модуля Wallet інженер імітує поповнення балансу гаманця через тестові платіжні шлюзи, перевіряючи реакцію системи на суми 19.99 та 20.00 G-pass.

Основними інструментами на цьому етапі є Swagger UI (для аналізу структури JSON-відповідей API) та Chrome DevTools (для моніторингу JavaScript-помилки у консолі та відстеження мережових запитів XHR/WS). Саме на Dev-сервері проводиться інтенсивне негативне тестування, імітуються збої клієнт-серверного з'єднання та перевіряється ідемпотентність транзакцій.

Ключовим фактором надійності системи тестування GoSeeChat є повна функціональна та конфігураційна ідентичність обох середовищ. З погляду комп'ютерної інженерії, це підтримується дзеркальним налаштуванням серверів, включаючи версії API, схеми реляційної бази даних та параметри політики безпеки CORS. Така точність гарантує, що поведінка програмного забезпечення, верифікована в тестовому контурі, репродукується на промисловому сервері без виникнення дефектів оточення.

Процес перенесення програмних компонентів між середовищами складається з чотирьох послідовних етапів:

- розробка та ізольоване тестування: Розробники розгортають вихідний код нової функціональності на Dev-сервері;
- верифікація в тестовому середовищі: QA-інженер виконує відповідні сценарії із «Checklist GoSeeChat», реєструючи виявлені дефекти у системі управління Jira [5];
- регресійне тестування (Regression Testing): Після усунення дефектів розробниками проводиться повторна перевірка виправлень та взаємопов'язаних модулів на Dev-сервері;
- реліз на Prod-сервер: За умови успішного підписання звіту про якість (Release QA Sign-off), гілки коду об'єднуються з основною лінією розробки та розгортаються на промисловому сервері.

Prod-сервер — це чинне промислове середовище, де перебувають реальні Продюсери, Креатори та Клієнти, а 1 одиниця внутрішньої валюти G-pass суворо еквівалентна номіналу в 1 Євро. Тестування на цьому рівні має виключно приймальний характер (Acceptance/Smoke Testing). Його завданням є підтвердження успішності процесу розгортання, валідація синхронізації бази даних та перевірка працездатності критичних шляхів користувача (Critical Paths). Одразу після завершення деплою QA-інженер проводить фінальну верифікацію на Prod-сервері за скороченим списком ключових параметрів:

- автентифікація: Успішний вхід до системи через реальні профілі Google або Apple ID.

– модуль Wallet: Коректне відображення поточного балансу гаманця та завантаження історичних логів платежів.

– комунікація та тарифікація: Написання текстового повідомлення в реальному часі та перевірка відпрацювання алгоритму тарифікації (з урахуванням ліміту у 280 символів).

На Prod-сервері тестові операції виконуються з максимальною обережністю для запобігання генерації надлишкових («сміттєвих») даних або випадкового списання реальних безготівкових коштів. Будь-які девіації інтерфейсу на цьому етапі стають миттєво видимими для всієї аудиторії платформи, що підвищує рівень відповідальності та критичності цієї стадії контролю.

Порівняльний аналіз процедур і параметрів тестування на Dev- та Prod-серверах наведено в таблиці 3.4.

Таблиця 3.4 — Порівняльний аналіз процедур тестування в розподілених середовищах платформи GoSeeChat

Параметр порівняння	Тестове середовище (Dev-сервер)	Промислове середовище (Prod-сервер)
Основна інженерна мета	Глибоке налагодження коду, діагностика помилок бізнес-логіки (Debug/Functional Testing).	Валідація успішності деплою, перевірка працездатності критичних функцій (Smoke/Acceptance Testing).
Тип використовуваних даних	Синтетичні, тестові дані (Mock-об'єкти, тестові платіжні картки).	Реальні дані користувачів, фактичні платіжні рахунки та транзакційні записи.
Основний інструментарій	Swagger UI, інструменти розробника Chrome DevTools, логи сервера, СУБД додатка.	Верифікація через графічний інтерфейс користувача, моніторинг промислових логів.

Продовження таблиці 3.4

Параметр порівняння	Тестове середовище (Dev-сервер)	Промислове середовище (Prod-сервер)
Рівень технічних ризиків	Мінімальний. Помилки коду не впливають на користувачів та стабільність бізнесу.	Критичний. Дефекти призводять до фінансових втрат та погіршення репутації продукту.
Доступ до середовища	Закритий контур (доступ обмежений внутрішньою мережею команди розробки).	Публічний контур (відкритий доступ для всіх категорій користувачів інтернету).

Розподілена структура тестової інфраструктури GoSeeChat дозволяє сформуванню ефективний багатоступеневий фільтр програмних дефектів. Dev-сервер забезпечує умови для проведення глибокої технічної діагностики та верифікації бізнес-логіки з використанням інструментів Swagger UI та Chrome DevTools, тоді як Prod-сервер гарантує фінальну перевірку працездатності продукту в реальних умовах промислової експлуатації. Такий інженерний підхід дозволяє локалізувати понад 95% логічних помилок на етапі розробки, що забезпечує належну фінансову безпеку та стабільність соціальної платформи для кінцевого споживача.

4. АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

4.1 Аналіз виявлених дефектів, регресія та статистика виправлень

Регресійне тестування виступає основним інструментом перевірки цілісності програмного комплексу GoSeeChat після внесення змін до вихідного коду, усунення дефектів або рефакторингу. Головна інженерна мета цього етапу полягає в підтвердні того, що модифікація окремих модулів не спричинила

регресії — тобто появи нових помилок у раніше верифікованих функціональних вузлах [15].

В межах кваліфікаційного дослідження аналіз результатів охоплює повний цикл випробувань для трьох базових бізнес-ролей: Продюсера, Креатора та Клієнта. Зіставлення планових показників структурованого логу та фактичних результатів виконання тест-кейсів дозволило сформувати об'єктивну статистичну картину поточної якості десктопної версії вебплатформи.

Для математичного обґрунтування стабільності клієнт-серверної взаємодії та оцінки надійності системи розраховано ключові метрики ефективності тестування. На основі фактичних даних повного регресійного пулу загальний обсяг виконаних тест-кейсів склав 298 одиниць. Статистичні дані розподілу результатів із фіксацією успішно пройдених (PASSED), зафіксованих дефектів (FAILED) та нереалізованих або пропущених через блокуючі баги сценаріїв (NOT PASS) відображено в таблиці 4.1.

Таблиця 4.1 — Зведена статистика виконання регресійного циклу GoSeeChat

Об'єкт дослідження (Бізнес-роль)	Заплановано тестів (шт.)	Виконано успішно (PASSED)	Виявлено дефектів (FAILED)	Блоковано / Не пройдено (NOT PASS)	Коефіцієнт успішності (Kpass, %)	Сумарна QA Estimate
Клієнт (Client)	148	134	5	6	90.54 %	212 хв
Креатор (Creator)	94	86	5	1	91.49 %	154 хв
Продюсер (Producer)	56	49	8	0	87.50 %	118 хв
Усього по системі	298	271	18	9	90.9 %	484 хв

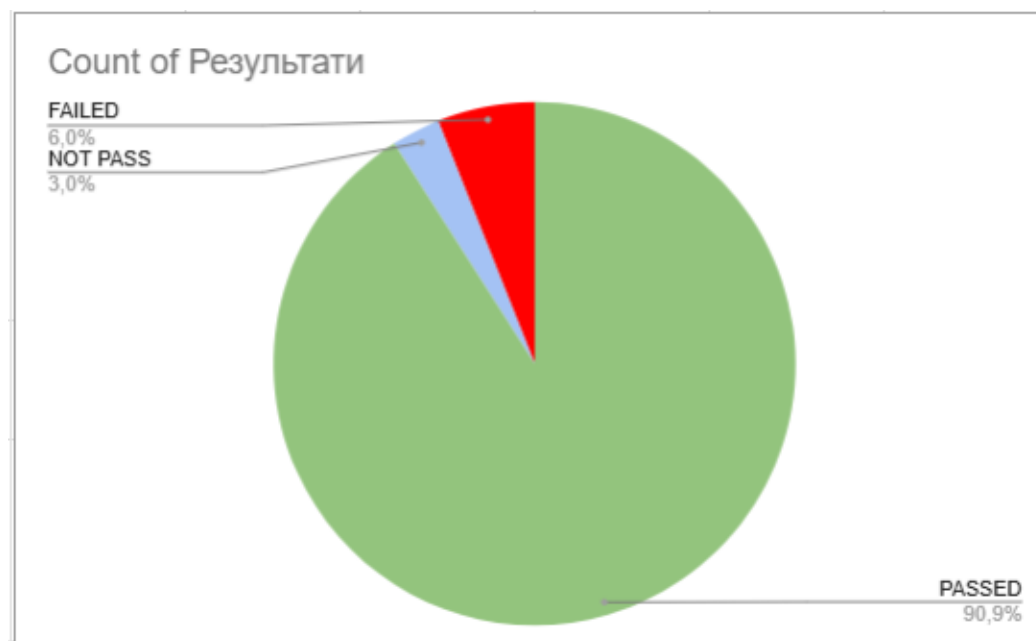


Рисунок 4.1 — Діаграма виконання регресійного циклу GoSeeChat

Аналіз обчислень показує, що загальний коефіцієнт успішності виконання тестових сценаріїв, який становить 90.9%. Найнижчий показник зафіксовано в модулі Продюсера, що обумовлено підвищеною архітектурною складністю ієрархічних зв'язків та розрахункових таблиць у цьому ролі. Наявність 7 блокованих тест-кейсів (NOT PASS) у загальному пулі пов'язана з дефектами інтеграції платіжних шлюзів, які унеможливили проходження наскрізних сценаріїв.

Для візуалізації структури виявлених невідповідностей та оцінки їхнього впливу на готовність програмного продукту до промислової експлуатації всі 22 зафіксовані дефекти класифіковано за рівнями критичності відповідно до інженерних стандартів QA. Співвідношення категорій дефектів наведено на рисунку 4.2.

Висока питома вага дефектів рівнів Critical та Major вимагала негайного блокування відповідних гілок розробки до моменту повного усунення аномалій, оскільки ці помилки безпосередньо порушували фінансову безпеку, ідемпотентність транзакцій та логіку розмежування прав доступу між Клієнтом, Креатором і Продюсером.

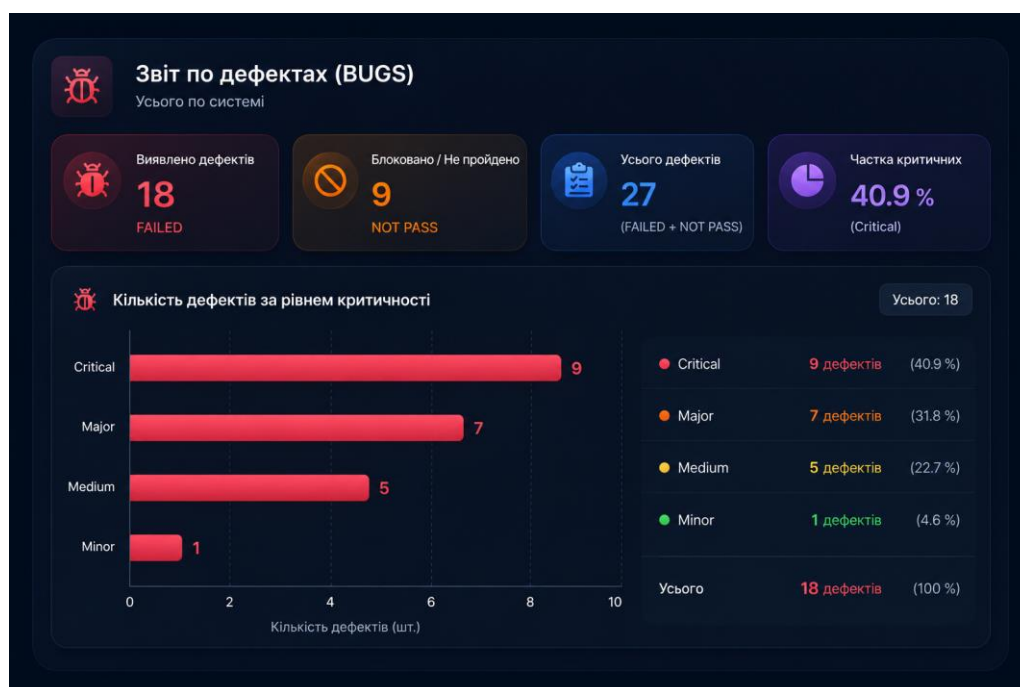


Рисунок 4.2 Структурний розподіл зафіксованих регресійних дефектів за ступенем критичності (Severity).

На основі протоколу регресійного циклу виконано деструктуризацію виявлених помилок за їхньою технічною суттю та місцем локалізації в архітектурі системи:

- локалізовано дефект у контекстному меню Клієнта, що блокував працездатність функцій «Hidden» та «Report» у мультимедійних розділах Video, Photo, Post. При виклику опції Hide Video медіаконтент не приховувався з інтерфейсу сторінки. Аналіз мережевого трафіку інструментами розробника зафіксував повну відсутність вихідних HTTP-запитів до API при натисканні на елементи керування, що свідчило про відсутність прив'язки обробників подій (Event Listeners) на рівні фронтенд-компонентів;
- виявлено критичне порушення логіки розмежування прав при взаємному блокуванні користувачів. Коли Креатор додавав Клієнта до чорного списку, Креатор помилково відображався у списку Blocked самого Клієнта. Це дозволяло заблокованому Клієнту самотійно ініціювати процедуру розблокування (Unban), відновлювати доступ до профілю Креатора та ігнорувати встановлені обмеження приватності, що повністю нівелювало захисну функцію блокування.

Для забезпечення можливості швидкого відтворення та подальшого усунення виявлених дефектів усі аномалії реєструвалися в системі управління завданнями Jira із суворим дотриманням технічного регламенту. До кожного баг-репорту додавалися унікальний ідентифікатор дефекту, рівень критичності, пріоритет, умови виникнення, сценарій відтворення, очікуваний та фактичний результати, а також верифікаційні матеріали (скріншоти, відеофіксація помилок, журнали системних подій і логи Chrome DevTools та Swagger UI).

Нижче наведено відображення зафіксованих дефектів, адаптовані під вимоги реєстрації інцидентів у системі Jira.

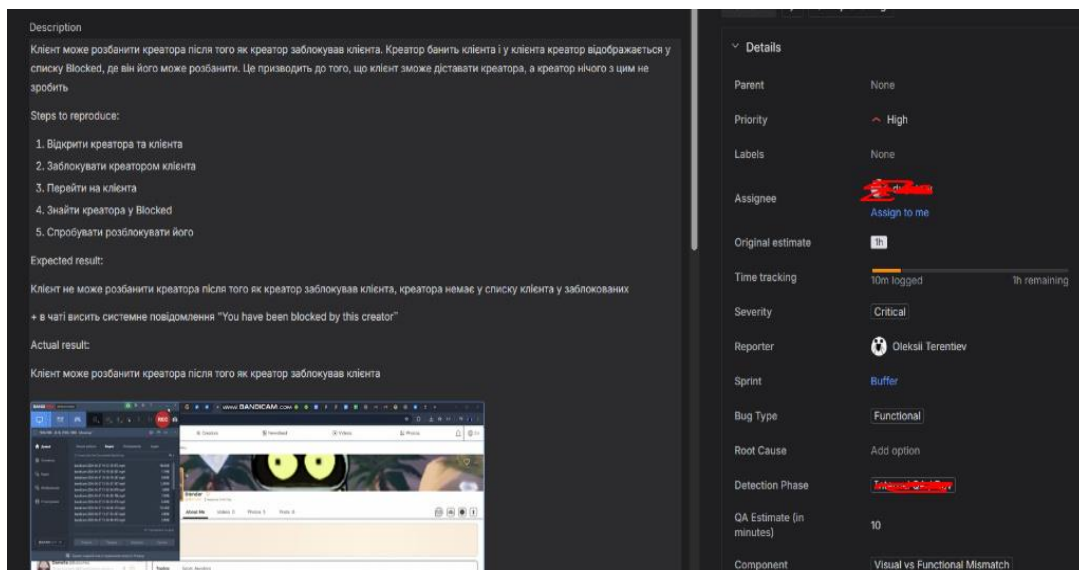


Рисунок 4.3 — Заведення багу GSC-6111 у систему Jira

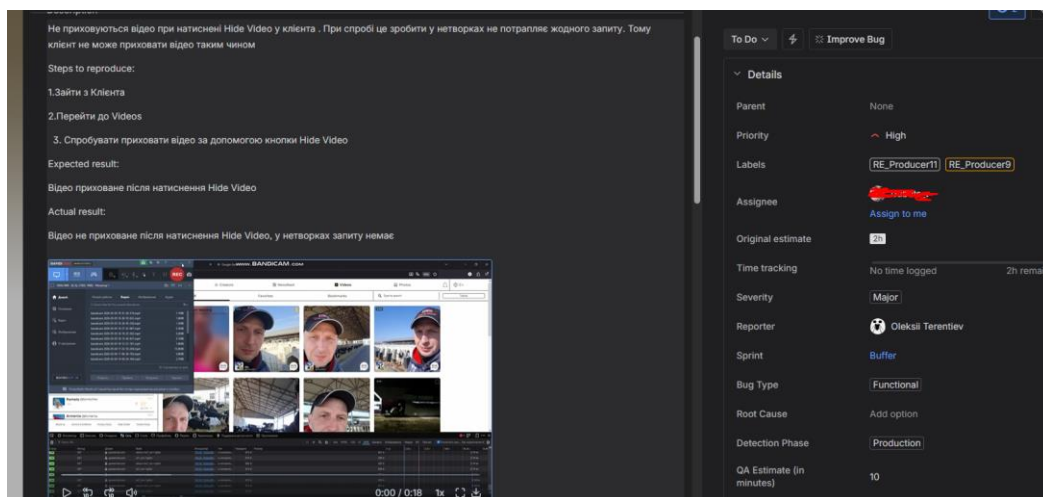


Рисунок 4.4 - Заведення багу GSC-6206 у систему Jira

Усі зареєстровані дефекти пройшли повний життєвий цикл обробки відповідно до методології розробки програмного забезпечення:



Рисунок 4.5 – Життєвий цикл багу

Після отримання оновлених збірок виконувалось повторне тестування (Re-testing), а також регресійна перевірка суміжних модулів.

Контроль коректності внесених змін здійснювався із застосуванням:

- Chrome DevTools (вкладка Network);
- Swagger UI;
- аналізу серверних запитів;
- перевірки журналів виконання.

Інструментальний контроль за допомогою засобів розробника Chrome DevTools (вкладка Network) та Swagger UI підтвердив успішність внесених виправлень:

- для дефекту контекстного меню Клієнта: Розробники відновили логіку обробки подій кліку на елементах контекстного меню «Hidden» та «Report». У фронтенд-архітектуру інтегровано тригери для генерації та відправки відповідних POST/PATCH запитів до API. Інструментальний контроль підтвердив, що при виклику команди Hide Video система стабільно надсилає мережевий запит, а контент миттєво видаляється з активного вікна відображення;
- для дефекту логіки взаємоблокування: Повністю перероблено логіку фільтрації користувачів у базі даних на рівні бекенду. Забезпечено суворе розмежування статусів: якщо Креатор блокує Клієнта, Клієнт позбавляється технічної можливості впливати на статус блокування. При повторній верифікації

перевірено, що профіль Креатора більше не з'являється у списку заблокованих Клієнтом, а в робочій зоні чату відображається статичне системне повідомлення «You have been blocked by this creator». Схему інструментальної верифікації та логічних станів після оптимізації захисту відображено на рисунку;

Результати повторного тестування підтвердили успішне усунення дефектів GSC-6111 та GSC-6206 без виникнення побічних аномалій у суміжних функціональних компонентах системи.

ВИСНОВКИ

У цій дипломній роботі розв'язано актуальну науково-практичну задачу, що полягає в оптимізації процесу забезпечення якості та підвищенні стабільності функціонування високонавантаженої соціальної платформи GoSeeChat. Усі результати інженерних випробувань та аналітичних розрахунків повністю відповідають вимогам технічного завдання.

Проведені дослідження, інфраструктурні перетворення та цикли тестування дозволяють сформулювати такі основні висновки:

Обґрунтовано та впроваджено архітектурний принцип розподілу інфраструктурних контурів. Відокремлення тестового середовища (Dev-сервер) від промислової експлуатації (Prod-сервер) забезпечило ізольовану верифікацію програмного коду. Створення дзеркальних конфігурацій серверів, включаючи ідентичність версій API, схем реляційних баз даних та налаштувань CORS, усунуло появу дефектів оточення під час розгортання оновлень на робочому сервері.

Забезпечено наскрізне покриття функціональності для базових бізнес-ролей. У межах регресійного циклу сформовано та виконано структурований пул із 298 тест-кейсів. Тестові сценарії охопили критичні шляхи користувачів для трьох ключових суб'єктів платформи: Клієнта (148 сценаріїв), Креатора (94 сценаріїв) та Продюсера (56 сценаріїв). Це дозволило оцінити стабільність клієнт-серверної взаємодії та функціональну працездатність основних бізнес-процесів системи.

Проведено статистичний аналіз результатів регресійного тестування та оцінено ефективність виконання тестових сценаріїв. Загальний коефіцієнт успішності виконання тестів склав 90.9%, при цьому із 298 запланованих сценаріїв успішно виконано 271 тест-кейс, виявлено 18 дефектів, а 9 сценаріїв отримали статус NOT PASS. Найнижчий показник успішності зафіксовано у модулі Продюсера (87.50%), що пояснюється підвищеною складністю ієрархічної структури функціональних зв'язків та механізмів фінансових розрахунків.

Загальний час виконання регресійного циклу становив 484 хвилини, що дозволило сформулювати кількісну оцінку трудомісткості процесу тестування.

Проведено класифікацію та локалізацію дефектів відповідно до їх функціональної природи та місця виникнення в архітектурі системи. У процесі тестування було локалізовано дефект у контекстному меню Клієнта, який блокував працездатність функцій Hidden та Report у мультимедійних розділах Video, Photo та Post. Аналіз мережевого трафіку за допомогою Chrome DevTools встановив відсутність генерації HTTP-запитів при виклику функцій приховування контенту, що свідчило про порушення механізму прив'язки обробників подій у фронтенд-компонентах системи.

Виявлено та усунуто критичне порушення логіки розмежування прав доступу між користувачами системи. Під час тестування механізму взаємного блокування встановлено, що при додаванні Клієнта до списку заблокованих користувачів Креатором система помилково відображала Креатора у списку Blocked самого Клієнта. Це дозволяло заблокованому користувачу ініціювати процедуру розблокування та отримувати повторний доступ до профілю Креатора, що створювало потенційний ризик порушення політики приватності. Для усунення проблеми було модифіковано механізм фільтрації записів на серверній стороні та реалізовано суворе розмежування логічних станів користувачів.

Оцінено ефективність проведених оптимізацій та практичну цінність отриманих результатів. Використання системи реєстрації дефектів Jira, засобів аналізу Chrome DevTools та Swagger UI дозволило забезпечити швидку локалізацію, документування та усунення виявлених аномалій. Під час повторного тестування підтверджено коректність внесених змін: функції контекстного меню почали формувати відповідні API-запити, а механізм взаємного блокування користувачів виключив можливість обходу встановлених обмежень доступу. Результати повторної верифікації підтвердили відсутність побічних дефектів у суміжних функціональних модулях системи та підвищення загального рівня стабільності програмного продукту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. На заміну ДСТУ 3008-95 ; чинний від 2017-07-01. Вид. офіц. Київ : УкрНДНЦ, 2016. 32 с.
2. ДСТУ 8302:2015. Інформація і документація. Бібліографічне посилання. Загальні положення та правила складання. Чинний від 2016-07-01. Вид. офіц. Київ : УкрНДНЦ, 2016. 30 с.
3. Нові бланки НУ «Запорізька політехніка». Запорізька політехніка. URL: <https://zp.edu.ua/novi-blanki-zntu/> (дата звернення: 12.03.2026).
4. Соціальна платформа **GoSeeChat**: офіційний сайт. URL: <https://goseechat.com> (дата звернення: 15.03.2026).
5. Atlassian Jira Software Cloud: Documentation for Test Management, Bug Tracking Workflows and Agile Project Management. Atlassian Support. 2026. URL: <https://support.atlassian.com/jira-software-cloud/> (date of access: 18.03.2026).
6. Swagger UI OpenAPI Specification Tool for Interactive API Documentation and Functional Testing. SmartBear Software. 2025. URL: <https://swagger.io/tools/swagger-ui/> (date of access: 22.03.2026).
7. Chrome DevTools Network Activity Reference: Analyzing HTTP, XHR and WebSocket Request Traffic. Google Developers. 2025. URL: <https://developer.chrome.com/docs/devtools/network/> (date of access: 25.03.2026).
8. GoSeeChat Platform Business Requirements Specification (BRS). Version 2.4. Internal Documentation. GoSeeChat Development Team, 2025. 687 p.
9. Результати виконання регресійного пулу тест-кейсів: Таблиця «Regression testing Goseechat». Звітні матеріали забезпечення якості (QA Checklists). URL: <https://docs.google.com/spreadsheets/d/13hJVkHRtIutJaPRIp4popKJto6G6QjMBFSZu6rId6NU/edit?gid=1158421419#gid=115842141>.
10. Баранцев О. В. Основи ручного тестування програмного забезпечення (Manual QA): Тест-дизайн та інструменти аналізу мережевих запитів : підруч. Львів : Новий світ – 2000, 2024. 256 с.

11. Certified Tester Foundation Level Syllabus v4.0. International Software Testing Qualifications Board (ISTQB). 2023. URL: <https://www.istqb.org/certifications/certified-tester-foundation-level> (date of access: 28.03.2026).

12. Коротєєва Т. О. Тестування програмного забезпечення : навч. посіб. Львів : Видавництво Львівської політехніки, 2022. 184 с.

13. Поворознюк А. І., Клімова О. С. Методи та засоби контролю якості веб-орієнтованих програмних систем : монографія. Харків : ХНУРЕ, 2023. 152 с.

14. OWASP Top 10:2021. The Ten Most Critical Web Application Security Risks. Open Web Application Security Project. URL: <https://owasp.org/www-project-top-ten/> (date of access: 04.04.2026).

15. Савченко В. В. Управління якістю програмного забезпечення в Agile-проектах : навч. посіб. Київ : КНЕУ, 2024. 210 с.

ДОДАТОК А
СЛАЙДИ ПРЕЗЕНТАЦІЇ

Національний університет «Запорізька політехніка»
Кафедра комп'ютерних систем та мереж | Спеціальність 123 — Комп'ютерна інженерія

РОЗРОБКА СИСТЕМИ ТЕСТУВАННЯ
СОЦІАЛЬНОЇ ПЛАТФОРМИ

GoSeeChat

Виконав: студент гр. КНТ-522 Терент'єв Олексій Сергійович
Керівник: к.т.н., доцент Тягунова М.Ю.

Запоріжжя — 2026

Рисунок А.1 – Слайд 1

МЕТА ТА ОБ'ЄКТ ДОСЛІДЖЕННЯ

ОБ'ЄКТ

GoSeeChat — соціальна FinTech-платформа для обміну цифровим контентом із вбудованою системою мікротранзакцій. Платформа поєднує функції соціальної мережі, цифрового маркетплейсу та платіжного сервісу з триольовою моделлю доступу.

Клієнт

Споживач контенту. Поповнює баланс G-pass, веде чати з Креаторами, керує приватністю профілю

Креатор

Автор контенту. Проходить KYC-верифікацію, завантажує медіа в Mediabank, встановлює ціни та розсилки

Продюсер

Управлінець команди. Налаштовує розподіл доходу (правило 100%), модерує Креаторів, контролює фінанси

Внутрішня валюта: 1 G-pass = 1 Євро

МЕТА

Розробка та впровадження комплексної системи ручного тестування (Manual QA) платформи GoSeeChat, яка забезпечує повну верифікацію бізнес-логіки, фінансових механізмів та захисту персональних даних користувачів.

01

Верифікація бізнес-логіки взаємодії Клієнта, Креатора та Продюсера

02

Перевірка точності фінансових транзакцій та ідемпотентності операцій G-pass

03

Тестування механізмів безпеки, KYC-верифікації та модерації контенту

04

Контроль адаптивності GUI у діапазоні 390px–1920px через Dev/Prod середовища

05

Локалізація дефектів за допомогою Swagger UI та Chrome DevTools

Рисунок А.2 – Слайд 2

АКТУАЛЬНІСТЬ ТА ВСТУП

Глобальні соціальні медіа стрімко трансформуються зі звичайних засобів комунікації у багатофункціональні FinTech-екосистеми. Сучасні платформи інтегрують функціонал соціальних мереж, цифрових маркетплейсів та платіжних сервісів, дозволяючи монетизувати інтелектуальну власність. Технічна якість ПЗ напряду визначає ринковий успіх — будь-які дефекти бізнес-логіки призводять до критичних репутаційних та фінансових втрат компанії.

Проблема: GoSeeChat оперує конфіденційними даними та фінансовими потоками (G-pass = 1 Євро) — помилки в алгоритмах тарифікації, розподілу прибутків або приватних режимів генерують критичні ризики безпеки та стабільності бізнесу

Чому ручне тестування (Manual QA)?

Специфіка GoSeeChat унеможливує використання виключно автоматизованих рішень. Валідація нелінійних сценаріїв вимагає методів Manual QA: верифікація профілів (KYC), модерація медіаконтенту категорій «18+» та «Sensitive», контроль виконання угод SLA щодо оперативності відповідей Креаторів. Manual QA забезпечує гнучкість, необхідну для імітації реальної поведінки користувачів за умов динамічних бізнес-вимог — критично для GoSeeChat, де черговість видачі контенту та алгоритми розсилки залежать від багатьох змінних параметрів.

Тарифікація повідомлень Межа 280 символів: ≤280 — стандартний платний блок (60 сек. вікно), ≥281 — окремий преміальний блок з фіксованою ставкою 3.49 Євро	KYC та верифікація особистості Перед монетизацією — завантаження фото документів (до 15 Мб) та відеоідентифікація з промовлянням контрольної фрази
Модерація 18+/Sensitive Кожне порушення маркування = 1 штрафний бал; при накопиченні 5 балів — статус «На обліку»; при 10 балах — автоблокування акаунту	Newsletter Engine / Rockstar Сегментація COLD/WARM; алгоритм Rockstar підвищує пріоритет Креатора у пошуку на основі Response Time та активності відповідей

Рисунок А.3 – Слайд 3

РОЗДІЛ 1 — АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

GoSeeChat — складний програмний комплекс клієнт-серверної архітектури, що інтегрує елементи фінансової платформи (FinTech) та системи управління контентом (CMS). GUI стабільно функціонує у діапазоні 390×844 рх (мобільний) — 1920×1080 рх (десктоп). Система декомпонована на три базові функціональні модулі, взаємодія яких визначає всю бізнес-логіку платформи:

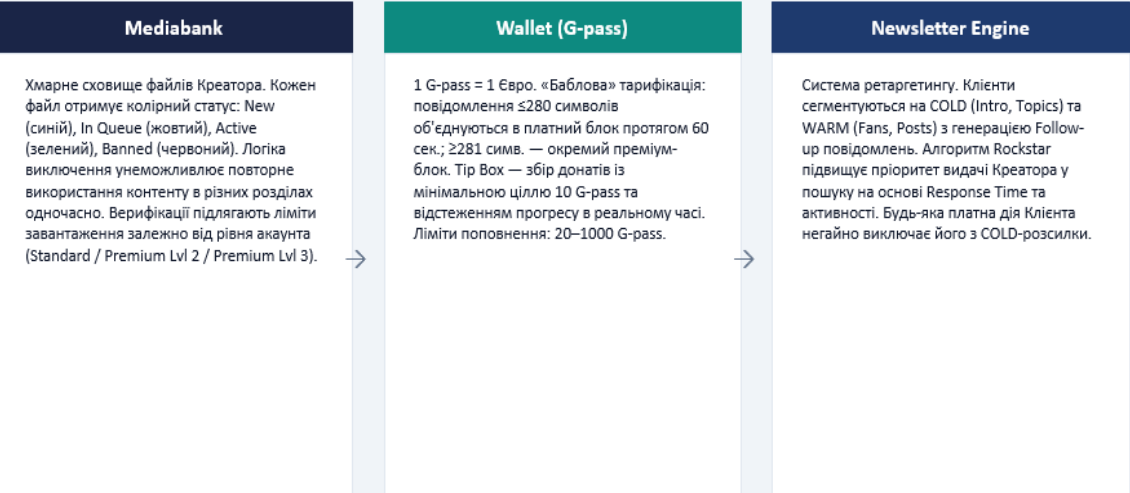


Рис. 1.1 — Схема взаємодії модулів Mediabank → Wallet → Newsletter Engine

Рисунок А.4 – Слайд 4

РОЗДІЛ 1 — РОЛЬОВА МОДЕЛЬ ТА ЛІМІТИ СИСТЕМИ

Таблиця 1.1 — Порівняльна характеристика прав доступу ролей GoSeeChat

Функціонал	Продюсер	Креатор	Клієнт
Редагування профілю	Ні	✓	✓
Фінансова звітність	Повна	Лише власна	Лише баланс
Управління Newsletter	✓	✓	Ні
Виведення коштів (Payout)	✓	✓	Ні

Таблиця 1.2 — Системні ліміти для різних рівнів акаунтів

Параметр	Standard	Premium Lvl 2	Premium Lvl 3
Публікацій на добу	3	10	∞ Безліміт
Об'єм Mediabank (Гб)	40	80	160
Активних нотаток (Notes)	8	16	64

⚠ Правило Продюсера: частка Продюсера + частка Креатора = 100% на рядок. Частка Креатора захищена від від'ємних значень — система блокує некоректне введення.

Авторизована зона платформи надає повноцінний доступ до Creators / NewsFeed / Videos / Photos / Streams. Блок «My Messages» розділений на вкладки GoSee (нові контакти) та Contacts (діалоги з платними взаємодіями) — ключовий індикатор монетизації.

Рисунок А.5 – Слайд 5

РОЗДІЛ 1 — МЕТОДОЛОГІЯ ТА ІНСТРУМЕНТАЛЬНЕ ЗАБЕЗПЕЧЕННЯ

Цикл тестування GoSeeChat (STLC) — рис. 2.1

01
Аналіз BRS
Специфікація вимог

02
Тест-дизайн
EP + BVA сценарії

03
Dev-сервер
Системне тестування

04
Перспекія
Staging контур

05
Prod-сервер
Smoke / Acceptance

2.4 — Загальна структура та алгоритм функціонування платформи (рис. 2.3, 2.4)

Tier 1 | Presentation (Frontend)

Tier 2 | Application (Backend)

Tier 3 | Data (Database)

Рис. 2.4 — Алгоритм обробки події: GUI → JWT перевірка → бізнес-модуль → БД → JSON 200 OK → перерендер UI

Рисунок А.6 – Слайд 6



Рисунок А.7 – Слайд 7



Рисунок А.8 – Слайд 8

РОЗДІЛ 3 — ТЕСТУВАННЯ РОЛІ КЛІЄНТ (3.1.3) | 148 ТЕСТ-КЕЙСІВ

Таблиця 3.3 — Фрагмент чек-ліста для бізнес-ролі Client

ID	Секція	Перевірка	Очікуваний результат	Estimate
CL-LOG-01	Log In	OAuth 2.0 (Google/Apple ID)	Сесія створена, редирект в роль	3 год
CL-ONB-01	Topics (Онбординг)	Вибір обов'язкових інтересів	Кнопка «Thank you» активна лише після мін. 1 вибору	2 год
CL-WLT-01	Wallet	Мін. поповнення ≥20 G-pass	Редирект на Verotel, статус In Process	5 год
CL-CHT-04	Chat	Повідомлення 281 символ	Списання за преміальним тарифом Msg In 281+	2 год
CL-VCH-02	Video Chat	3-сек. ліміт скасування	Cancel активний 3 сек., на 4-й запускає тарифікацію	3 год
CL-PRM-01	Premium	Upgrade до Level 3	Ліміт нотаток: 8→112 без релоаду сторінки	10 год
CL-SET-03	Privacy	Hide My Media toggle	Медіафайли блокуються для не-контактів (403 Forbidden)	5 год

Лоріка GoSee ↔ Contacts:

≤3 активних чатів → вкладка GoSee (пошук нових Креаторів)

>3 активних чатів → вкладка Contacts (пріоритет наявних діалогів)

Рисунок А.9 – Слайд 9

РОЗДІЛ 4 — АНАЛІЗ РЕЗУЛЬТАТІВ РЕГРЕСІЙНОГО ЦИКЛУ



Таблиця 4.5 — Зведена статистика регресійного циклу GoSeeChat

Бізнес-роль	Заплановано	PASSED	FAILED	NOT PASS	Kpass	QA Estimate
Клієнт (Client)	148	134	5	6	90.54%	212 хв
Креатор (Creator)	94	86	5	1	91.49%	154 хв
Продюсер (Producer)	56	49	8	0	87.50%	118 хв
УСЬОГО	298	271	18	9	90.9%	484 хв

Розподіл дефектів за критичністю (Severity):



Рисунок А.10– Слайд 10

РОЗДІЛ 4 — КЛЮЧОВІ ДЕФЕКТИ ТА ЇХ УСУНЕННЯ	
GSC-6206 MAJOR Симптом: Виправлення: CLOSED ✓	Контекстне меню Клієнта: не працювали функції «Hidden» та «Report» При натисканні Hide Video медіаконтент не приховувався. Chrome DevTools зафіксував відсутність HTTP-запитів → відсутність Event Listeners на фронтенд-компонентах. Відновлено прив'язку обробників кліку → інтегровано тригери POST/PATCH-запитів → Hide Video формує запит, контент миттєво видаляється. Верифіковано через Network tab.
GSC-6111 CRITICAL Симптом: Виправлення: CLOSED ✓	Критичне порушення логіки взаємного блокування користувачів Коли Креатор блокував Клієнта — Креатор помилково відображався у списку Blocked самого Клієнта. Клієнт міг самостійно ініціювати Unban і відновити доступ до профілю Креатора. Перероблено фільтрацію на рівні бекенду БД. Профіль Креатора більше не з'являється у Blocked Клієнта. У чаті — статичне повідомлення «You have been blocked by this creator».

Рисунок А.11 – Слайд 11

ВИСНОВКИ	
1	Впроваджено розподіл інфраструктури Dev/Prod — дзеркальні конфігурації API, схем БД та CORS усунули дефекти оточення при деплої.
2	Сформовано 298 тест-кейсів, що охоплюють усі 3 ролі: Клієнт (148), Креатор (94), Продюсер (56). Покриття критичних шляхів 100%.
3	Загальний Kpass = 90.9%. Найнижчий показник у Продюсера (87.5%) через складні фінансові зв'язки. Сумарна трудомісткість: 484 хв.
4	Локалізовано та усунуто 18 дефектів (9 Critical, 7 Major, 5 Medium, 1 Minor), зокрема критичне порушення прав доступу при взаємоблокуванні.
5	Сформовано структуровану QA-методику для FinTech-медіаплатформ із мікротранзакціями — мінімізовано ризики тарифікаційних збоїв та витоків PII.
6	Практична цінність: Jira + Swagger UI + Chrome DevTools забезпечили повний цикл — від виявлення до верифікації виправлення дефектів.

Рисунок А.12 – Слайд 12