Пояснювальна записка
до курсової роботи

на тему: Веб-застосування онлайн трансляції медіаконтенту

КПІ.ІП-41.ХХХХХХ.02.81

Київ – 2025

ЗМІСТ

ВСТУП

Сфера споживання медіаконтенту є однією з найбільш динамічних галузей сучасної цифрової економіки. Веб-застосування для онлайн трансляцій залишаються актуальними завдяки глобальному переходу від завантаження файлів до потокової передачі даних, що забезпечує миттєвий доступ до інформації та розваг. Актуальність даної роботи зумовлена зростанням попиту на незалежні стрімінгові платформи, що поєднують високу якість обслуговування з гнучкістю налаштувань під конкретні потреби бізнесу чи освіти.

Світові тенденції у розробці медіа-сервісів спрямовані на використання адаптивних протоколів передачі даних, хмарних інфраструктур та сучасних стандартів кодування відео. Провідні компанії, такі як YouTube, Twitch, Netflix та Vimeo, демонструють, що веб-технології здатні забезпечити телевізійну якість зображення та інтерактивність у реальному часі для мільйонів користувачів одночасно.

Сучасний стан розробки систем онлайн трансляції характеризується широкою доступністю інструментів для обробки відео та веб-серверів. Однак існують проблеми, пов’язані з високою затримкою передачі даних, нестабільністю відтворення при поганому з’єднанні та складністю масштабування архітектури під високі навантаження.

Провідні фахівці у галузі пропонують рішення на основі алгоритмів адаптивного бітрейту, використання мереж доставки контенту та вдосконалених методів кешування сегментів. У межах даної роботи обрано напрям розробки власного веб-застосування для онлайн трансляцій з акцентом на стабільність відтворення, модульну архітектуру та реалізацію механізму адаптації якості відео під умови мережі користувача.

ПЕРЕЛІК УМОВНИХ ПОСИЛАНЬ

ABR	–	Adaptive Bitrate — технологія адаптивного бітрейту, що дозволяє автоматично змінювати якість відеопотоку в реальному часі залежно від пропускної здатності мережі користувача.
API	–	Application Programming Interface — набір визначень, протоколів та інструментів для розробки програмного забезпечення, що дозволяє різним компонентам системи взаємодіяти між собою.
AWS	–	Amazon Web Services — хмарна платформа від компанії Amazon, що надає обчислювальні потужності, сховища даних та сервіси доставки контенту.
Azure	–	Microsoft Azure — хмарна платформа від компанії Microsoft для розробки, виконання застосунків і зберігання даних.
CDN	–	Content Delivery Network — географічно розподілена мережева інфраструктура, що забезпечує швидку доставку контенту кінцевим користувачам шляхом кешування даних на серверах, наближених до клієнта.
H.264/ H.265	–	Стандарти стиснення відео (кодеки), що використовуються для зменшення обсягу даних при збереженні високої якості зображення (H.265 є більш ефективним наступником H.264).
HLS	–	HTTP Live Streaming — комунікаційний протокол для потокової передачі медіаданих, розроблений Apple, який розбиває відеопотік на невеликі фрагменти для передачі через HTTP.
MPEG-DASH	–	Dynamic Adaptive Streaming over HTTP — міжнародний стандарт адаптивної потокової передачі мультимедійних даних, аналог HLS, який не прив'язаний до конкретного виробника.
QoS	–	Quality of Service — якість обслуговування; набір технологій та метрик, що гарантують певний рівень продуктивності потоку даних (мінімізація затримок, втрат пакетів).

1 ПЕРЕДПРОЄКТНЕ ОБСТЕЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

Веб-застосування онлайн трансляції медіаконтенту (стрімінгові сервіси) відносяться до класу розподілених інформаційних систем, у яких основний акцент робиться на безперервну передачу потокових аудіо- та відеоданих у реальному часі або за запитом. Ключовою особливістю таких систем є необхідність забезпечення мінімальної затримки при збереженні високої якості відтворення, що досягається використанням протоколів прикладного рівня, таких як HLS (HTTP Live Streaming) або MPEG-DASH [1, с. 245–250].

Сучасна розробка стрімінгових рішень здійснюється з використанням спеціалізованих медіа-серверів (наприклад, Nginx з модулем RTMP, Wowza) та клієнтських технологій HTML5. Основні етапи розробки включають захоплення сигналу, транскодування даних за стандартами стиснення H.264/H.265, сегментацію потоку та його доставку кінцевому користувачеві [2, с. 112–118]. Знання предметної області охоплює розуміння принципів цифрової обробки сигналів, алгоритмів адаптивного бітрейту (ABR) для підлаштування під швидкість інтернету користувача, а також методів кешування контенту [3, с. 58–65].

Розробка таких систем значно спростилася завдяки стандартизації веб-технологій та появі потужних API (Media Source Extensions, WebRTC). Проте основними проблемами сучасного стану галузі є:

- висока чутливість до втрати пакетів у мережі;
- затримки трансляції, зумовлені буферизацією [4];
- складність синхронізації аудіо та відео потоків;
- значні вимоги до обчислювальних ресурсів при транскодуванні.

Можливі шляхи покращення ситуації — використання сучасних кодеків (AV1), впровадження архітектури мікросервісів для масштабування навантаження та використання CDN (Content Delivery Networks) для оптимізації маршрутів доставки даних [5, С. 310–315].

У межах даної роботи обрано напрям розробки веб-застосування для онлайн трансляцій на базі протоколу HLS, з акцентом на створенні адаптивного плеєра та модульної серверної архітектури для забезпечення стабільності трансляції.

1.2 Аналіз існуючих рішень

Проаналізуємо відоме на сьогодні забезпечення у даній області та технічні рішення, що допоможуть у реалізації веб-застосування для онлайн трансляції медіаконтенту. Далі будуть розглянуті готові програмні продукти та алгоритмічні рішення, що використовуються в індустрії стрімінгу.

1.2.1 Аналіз відомих програмних продуктів

Для аналізу відомих програмних продуктів обрано 2 найпопулярніші платформи: Twitch та YouTube (YouTube Live)

Twitch

Twitch - це відеострімінговий сервіс, що спеціалізується на тематиці комп'ютерних ігор, зокрема трансляціях геймплею та кіберспортивних турнірів, а також творчого контенту та спілкування у форматі «Just Chatting». Власником сервісу є дочірня компанія Amazon. Платформа поєднує трансляцію відео в реальному часі з інтерактивною системою чату, що дозволяє миттєво взаємодіяти зі стрімером. Основний акцент зроблено на мінімальну затримку, щоб реакція ведучого на повідомлення була максимально швидкою.

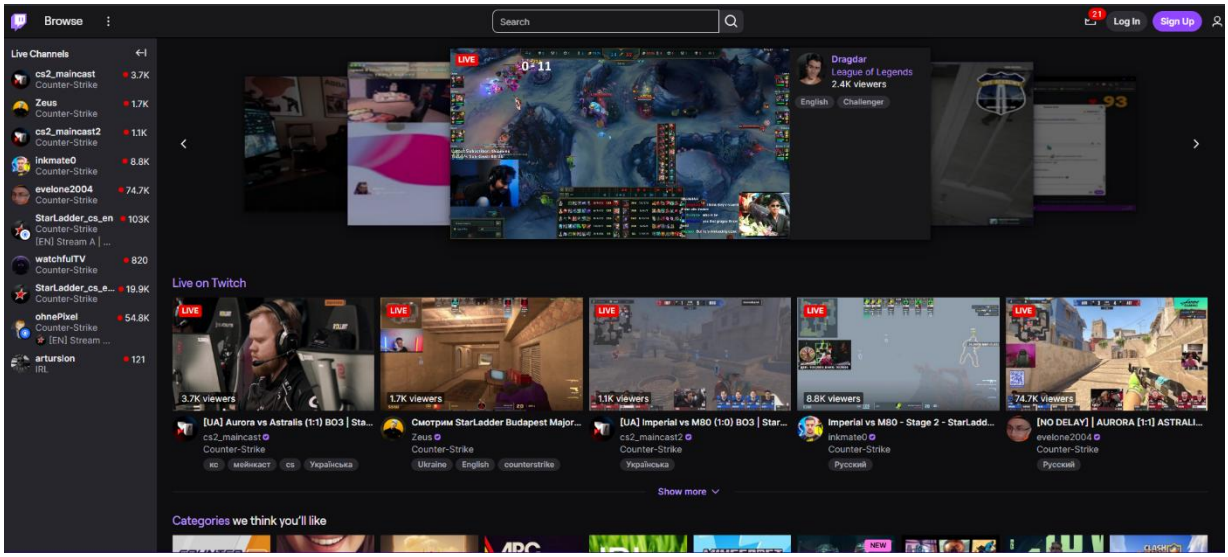

Рисунок 1.1 Інтерфейс головної сторінки у Twitch

Технічно система побудована на прийомі потоку через протокол RTMP (Real-Time Messaging Protocol) та його подальшій конвертації у формат HLS (HTTP Live Streaming) для доставки кінцевим користувачам. Відеоплеєр підтримує адаптивну зміну якості (Adaptive Bitrate Streaming), автоматично знижуючи роздільну здатність відео при погіршенні інтернет-з'єднання глядача.

Основний функціонал для користувача включає:

- Перегляд прямих ефірів з можливістю вибору якості (від 160p до 1080p60).
- Спілкування у текстовому чаті з використанням смайлів (Emotes).
- Систему підписок (Subscriptions) та донатів (Bits) для підтримки авторів.
- Перегляд записаних трансляцій (VOD) після завершення ефіру.

Twitch має суворі правила щодо авторського контенту, автоматично вимикаючи звук у збережених відео, якщо в них виявлено ліцензовану музику.

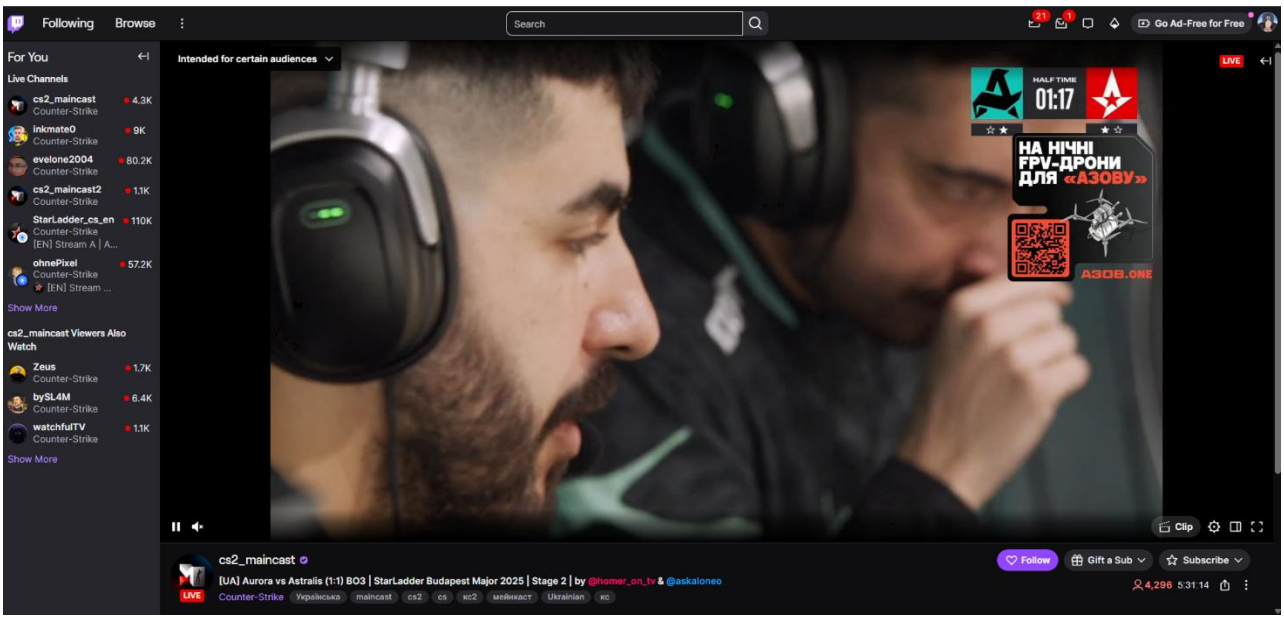

Рисунок 1.2 Інтерфейс перегляду трансляції на Twitch

YouTube

YouTube — це найбільший у світі відеохостинг, який також надає потужний функціонал для прямих трансляцій (YouTube Live). Сервіс належить компанії Google і використовує її глобальну інфраструктуру доставки контенту (CDN).

На відміну від Twitch, YouTube фокусується не лише на прямому ефірі, а й на збереженні контенту для довгострокового перегляду. Ключовою особливістю плеєра є функція DVR (Digital Video Recorder), яка дозволяє глядачеві перемотувати прямий ефір назад у будь-який момент, навіть якщо трансляція ще триває.

Архітектура YouTube дозволяє приймати вхідні потоки у високій роздільній здатності (до 4K та 8K) та підтримує кодеки VP9 та AV1, що забезпечують кращу якість зображення при меншому бітрейті порівняно зі стандартним H.264.

Функціональні можливості включають:

- Планування трансляцій із попереднім створенням сторінки очікування.
- Інтеграцію з екосистемою Google (авторизація, сповіщення).
- Аналітику в реальному часі для авторів (кількість глядачів, стан потоку).
- Автоматичне створення запису трансляції одразу після її завершення без необхідності додаткової обробки.

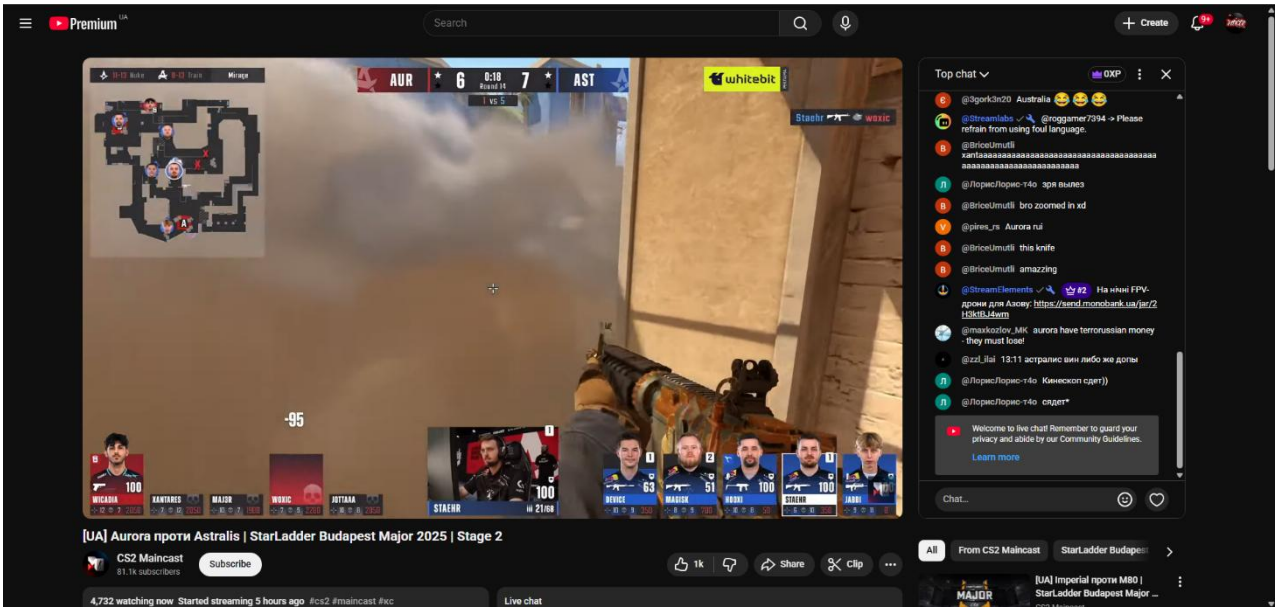

Рисунок 1.3 Інтерфейс прямої трансляції Youtube

Окрім технічних аспектів трансляції, платформа надає розширені інструменти для взаємодії з аудиторією та монетизації контенту, такі як Super Chat та спонсорство каналу. Управління ефіром здійснюється через YouTube Studio — спеціалізовану панель, що дозволяє модерувати чат, налаштовувати затримку потоку та відстежувати його технічні показники.

Проте, використання YouTube як платформи для спеціалізованих задач має свої недоліки: наявність сторонньої реклами, жорстка політика щодо авторського права, які можуть призвести до автоматичного блокування ефіру.

Для порівняння власної розробки з аналогами можна скористатись таблицею 1.1

Таблиця 1.1 – Порівняння з аналогами

Функціонал	Власна розробка	Twitch	Youtube	Пояснення
Використання протоколу HLS	+	+	+	Усі розглянуті системи використовують сегментовану передачу даних
Адаптивний бітрейт (ABR)	+	+	+	Можливість зміни якості відео залежно від швидкості мережі
Онлайн чат	+	+	+	Взаємодія глядачів у реальному часі є стандартом індустрії
Функція DVR (перемотування ефіру)	-	-	+	YouTube дозволяє перемотувати прямий ефір; Twitch та базова власна розробка — ні
Монетизація контенту	-	+	+	Власна розробка на даному етапі не передбачає платіжних систем
Збереження запису (VOD)	+	+	+	Можливість переглянути трансляцію після її завершення
Низька затримка (Low Latency)	+	+	+	Twitch та власна розробка пріоритезують швидкість; YouTube часто має більшу буферизацію заради якості

Продовження таблиці 1.1

Підтримка 4K трансляцій	-	-	+	Реалізація 4K вимагає значних серверних потужностей, доступних гігантам як Google

1.2.2 Аналіз відомих алгоритмічних та технічних рішень

Специфіка предметної області, пов’язана з передачею великих обсягів відеоданих, висуває особливі вимоги до стабільності з'єднання та пропускної здатності мережі. Ключовою проблемою розробки є забезпечення якості обслуговування (QoS) та мінімізація затримок (latency) в умовах нестабільного інтернет-з'єднання [5, с. 310–315].

Для аналізу існуючих рішень обрано два ключові завдання:

1. Алгоритми адаптації якості відео (Adaptive Bitrate Streaming — ABR).
2. Протоколи передачі потокових даних.

Алгоритми адаптації бітрейту (ABR)

Для забезпечення плавного відтворення без зупинок на буферизацію плеєр повинен динамічно змінювати бітрейт (якість) відео. В індустрії використовуються такі підходи до реалізації ABR:

a) Throughput-based (Засновані на пропускній здатності)

- Опис: Алгоритм оцінює швидкість завантаження попередніх сегментів відео та на основі цього прогнозує доступну смугу пропускання для наступних.
- Переваги: Швидка реакція на різке покращення умов мережі.
- Недоліки: Вразливість до короткочасних коливань швидкості (jitter), що може призвести до частих змін якості.
- Складність: O(1) для кожного сегмента [1, с. 250].

b) Buffer-based (Засновані на заповненості буфера)

- Опис: Рішення приймається на основі рівня заповнення буфера плеєра. Якщо буфер заповнений (наприклад, > 30 с) — якість підвищується, якщо майже порожній — знижується для уникнення зупинки.

– Переваги: Висока стабільність відтворення, менша ймовірність ребуферизації.

– Недоліки: Повільна реакція на початку відтворення (start-up phase) [3, с. 65].

c) Hybrid (Гібридні алгоритми)

– Опис: Комбінує оцінку пропускної здатності для швидкого старту та моніторинг буфера для стабілізації потоку в процесі перегляду.

Порівняння алгоритмів адаптації наведено у таблиці 1.2.

Таблиця 1.2 — Порівняння алгоритмів ABR

Алгоритм	Критерій прийняття рішення	Переваги	Недоліки
Throughput-based	Швидкість завантаження (Bandwidth estimation)	Швидкий старт, миттєва реакція	Нестабільність, обрізання якості
Buffer-based	Рівень буфера (Buffer occupancy)	Плавне відтворення, мінімум зупинок	Ризик низької якості на старті
Hybrid	Комбінація факторів	Оптимальний баланс (High QoE)	Висока складність реалізації

Оскільки для веб-застосування пріоритетом є стабільність відтворення, буде використано гібридний підхід, реалізований у сучасних клієнтських бібліотеках (наприклад, Hls.js), що дозволяє балансувати між якістю картинки та безперервністю потоку.

Протоколи передачі потокових даних

Вибір протоколу визначає затримку, сумісність із браузерами та складність серверної архітектури. Розглянемо основні стандарти:

a) RTMP (Real-Time Messaging Protocol)

– Застарілий стандарт на базі TCP, що забезпечує низьку затримку (3–5 с). Широко використовується для відправки потоку на сервер (Ingest), але не підтримується сучасними браузерами для відтворення [6].

b) HLS (HTTP Live Streaming)

– Протокол, що розбиває потік на короткі файли-сегменти (.ts) та створює плейлист (.m3u8). Працює через стандартний HTTP, що дозволяє використовувати кешування та CDN. Затримка зазвичай становить 10–30 с, але є стабільною [4].

c) WebRTC

– Забезпечує передачу даних у реальному часі (< 1 с) безпосередньо між браузерами або через медіа-сервер. Використовує UDP, що робить його вразливим до втрати пакетів при поганому з'єднанні [1, с. 255].

Порівняння протоколів наведено у таблиці 1.3.

Таблиця 1.3 — Порівняння протоколів передачі потокових даних

Протокол	Транспорт	Затримка (Latency)	Підтримка HTML5	Масштабованість
RTMP	TCP	Низька	Ні(потрібна конвертація)	Низька
WebRTC	UDP/TCP	Ультранизька (< 1 с)	Так	Складна
HLS	HTTP (TCP)	Середня (10+ с)	Так	Висока (CDN)

На основі аналізу для реалізації системи обрано:

1. Протокол HLS для доставки контенту глядачам, оскільки він забезпечує найкращу сумісність із веб-браузерами та дозволяє легко масштабувати навантаження через HTTP-кешування [4].
2. Стандарт стиснення H.264, оскільки він підтримується більшістю пристроїв на апаратному рівні, що знижує навантаження на процесор користувача під час декодування [2, с. 120].

1.3 Аналіз та моделювання бізнес-процесів

У межах розробки веб-застосування для онлайн трансляції медіаконтенту було визначено та детально змодельовано чотири ключові бізнес-процеси, що забезпечують функціонування системи: авторизація користувачів, обробка та трансляція потокового відео, адаптивне відтворення контенту на стороні клієнта та оформлення платної підписки.

Процес 1 – Авторизація користувача в системі.

Цей процес є фундаментальним для забезпечення безпеки платформи та персоналізації досвіду користувача. Його метою є верифікація особи користувача та

надання доступу до захищеного функціонала (панель керування стрімера, налаштування профілю). Процес спроєктовано з використанням дворівневої валідації даних (на клієнті та на сервері) для мінімізації навантаження на серверну частину. Для опису цього бізнес-процесу використовується BPMN 2.0 модель (рисунок 1.4).

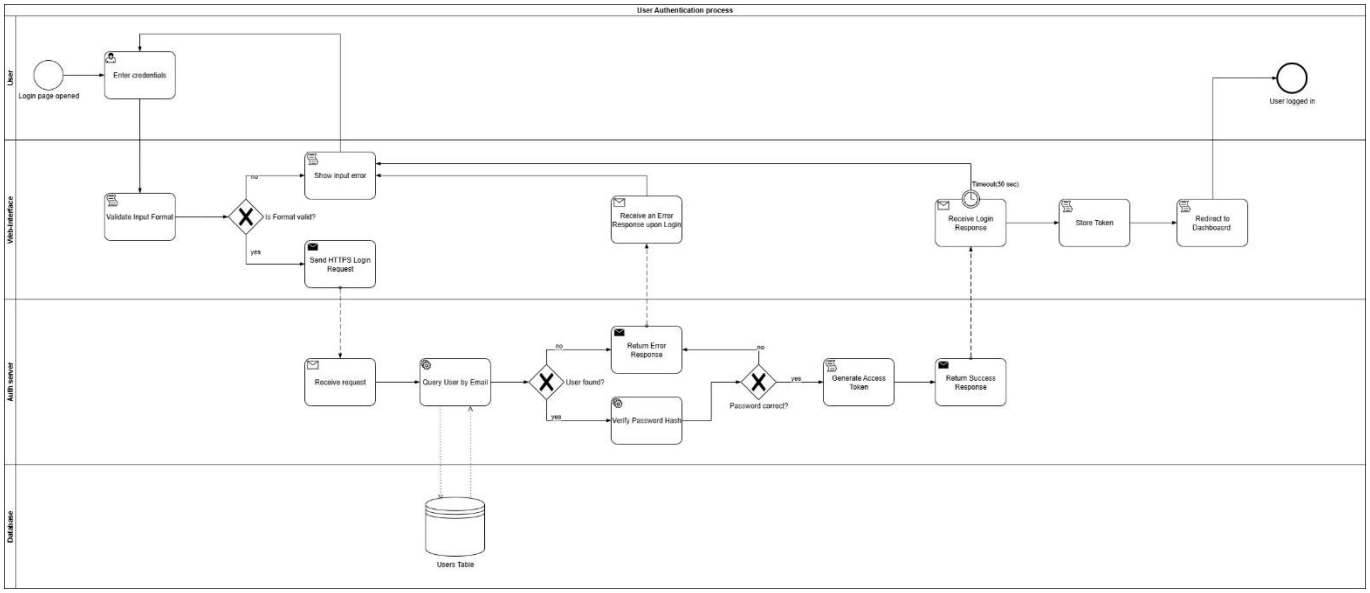

Рисунок 1.4 – Схема бізнес-процесу авторизації користувача

Посилання на діаграму в середовищі draw.io для кращого перегляду(4 діаграми послідовно розташовані в 1 файлі, їх потрібно відкрити за допомогою draw.io): https://drive.google.com/kpirgrdiagrams або відкрити фото у кращій якості з GitHub: https://github.com/blaz3xx/kpi2rgrdiagrams/tree/main

Опис процесу авторизації користувача:

– процес ініціюється відкриттям сторінки входу, де користувач вводить облікові дані (логін та пароль);

– на рівні веб-інтерфейсу відбувається первинна перевірка формату даних: якщо формат невірний, система миттєво відображає помилку, змушуючи користувача виправити введення без відправки запиту на сервер;

– після успішної валідації клієнт відправляє HTTPS-запит до сервера авторизації;

– сервер виконує пошук користувача в базі даних ("Users Table") за введеним email;

– якщо користувача знайдено, відбувається криптографічна звірка хешу пароля;

– у разі неспівпадіння даних або відсутності користувача, сервер повертає помилку, яка відображається в інтерфейсі;

– при успішній перевірці сервер генерує токен доступу (Access Token), відправляє його клієнту, після чого інтерфейс зберігає токен та перенаправляє користувача до особистого кабінету (Dashboard).

Процес 2 – Обробка та трансляція потокового відео (Live Streaming Pipeline). Даний бізнес-процес належить до категорії внутрішніх серверних операцій (Backend) і є критичним для функціонування сервісу. Він описує повний цикл перетворення вхідного RTMP-сигналу від стрімера у формат HLS, придатний для відтворення у браузерах.

Для опису цього бізнес-процесу використовується BPMN 2.0 модель (рисунок 1.5).

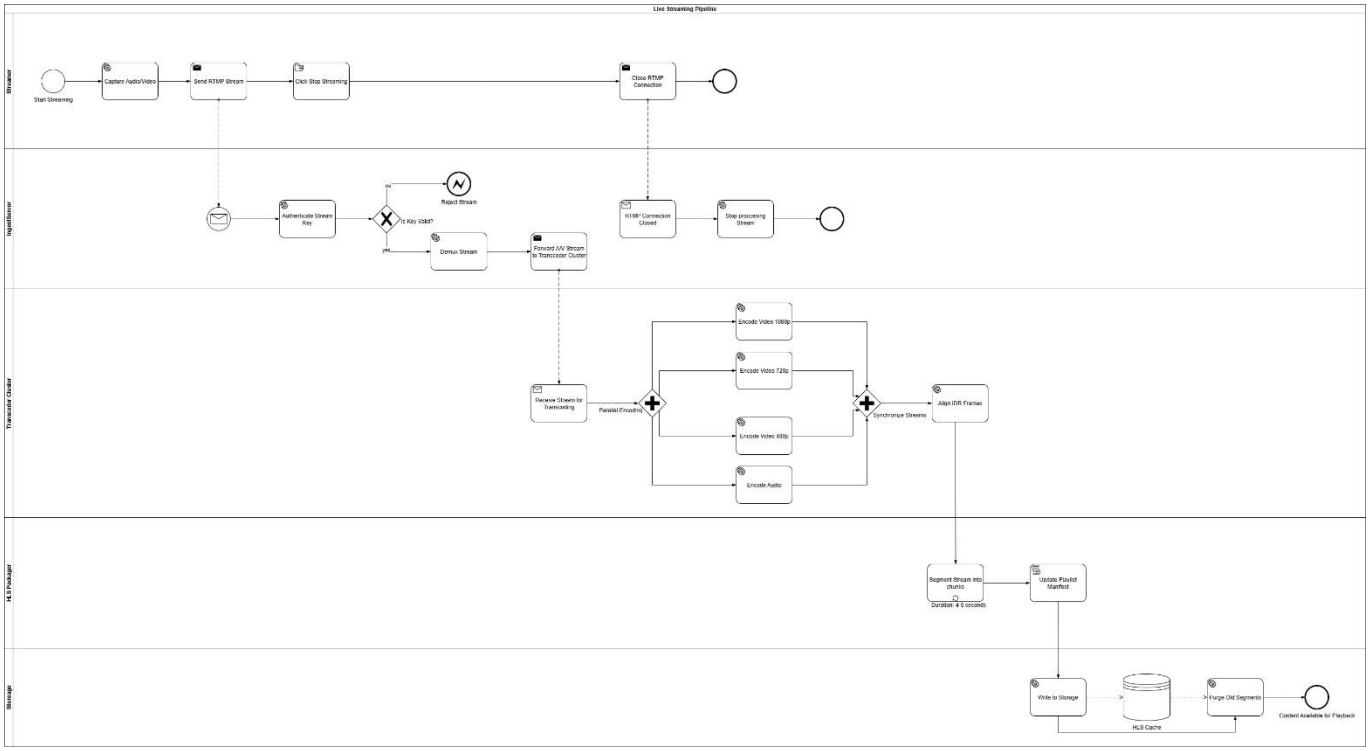

Рисунок 1.5 – Схема бізнес-процесу обробки відеопотоку

Посилання на діаграму в середовищі draw.io для кращого перегляду(4 діаграми послідовно розташовані в 1 файлі, їх потрібно відкрити за допомогою draw.io): https://drive.google.com/kpirgrdiagrams або відкрити фото у кращій якості з GitHub: https://github.com/blaz3xx/kpi2rgrdiagrams/tree/main

Опис процесу обробки відеопотоку:

– процес починається із запуску трансляції стрімером через програмне забезпечення (наприклад, OBS) та відправки RTMP-потоку на Ingest-сервер;

– система виконує аутентифікацію ключа трансляції (Stream Key): якщо ключ недійсний, з'єднання розривається;

– вхідний потік демультиплексується (розділяється на аудіо та відео) і передається у кластер транскодування;

– відбувається паралельне кодування відео у кілька роздільних здатностей (1080p, 720p, 480p) та обробка аудіо, після чого потоки синхронізуються та вирівнюються за ключовими кадрами (IDR frames);

– пакетувальник нарізає потоки на короткі сегменти (.ts файли тривалістю 4–6 секунд) та оновлює файл маніфесту (.m3u8);

– готові сегменти записуються у сховище (Storage/Cache) і стають доступними для завантаження глядачами;

– також запускається циклічний процес очищення застарілих сегментів для економії дискового простору, який триває до моменту зупинки трансляції стрімером.

Процес 3 – Адаптивне відтворення контенту (Client-Side Playback).

Цей процес описує логіку роботи відеоплеєра на стороні глядача. Його основна мета - забезпечити плавне відтворення відео без буферизації шляхом динамічної зміни якості зображення залежно від швидкості інтернету (Adaptive Bitrate Streaming).

Для опису цього бізнес-процесу використовується BPMN 2.0 модель (рисунок 1.6).

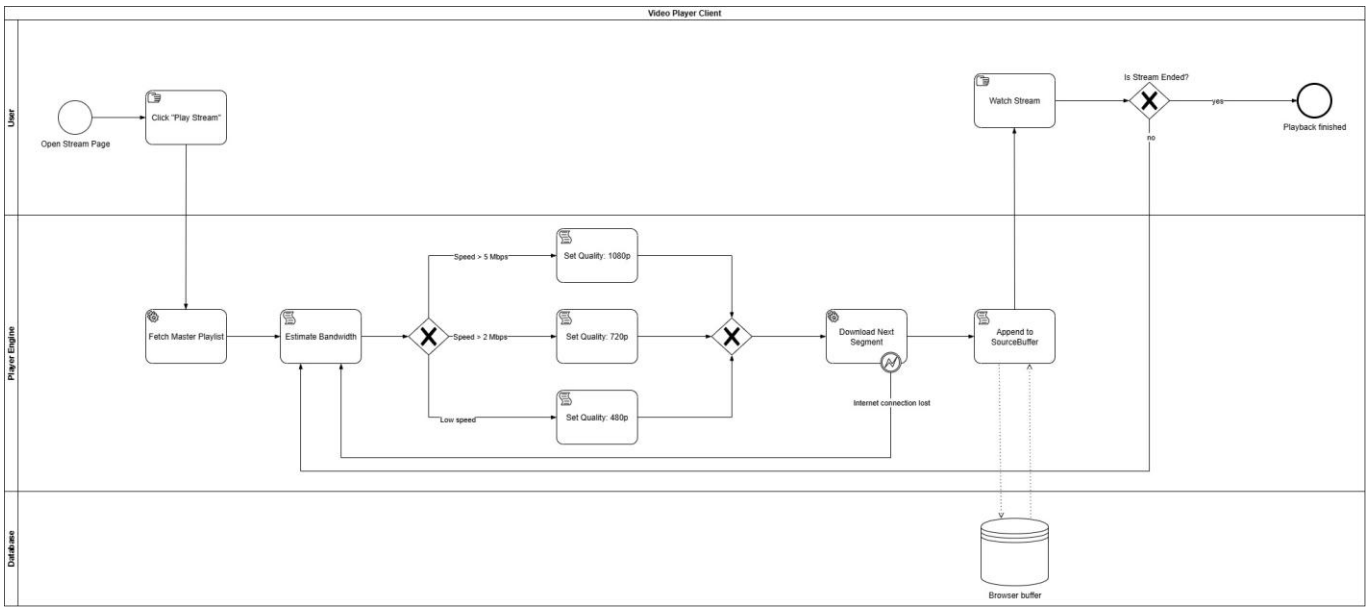

Рисунок 1.6 – Схема бізнес-процесу адаптивного відтворення стріму

Посилання на діаграму в середовищі draw.io для кращого перегляду(4 діаграми послідовно розташовані в 1 файлі, їх потрібно відкрити за допомогою draw.io): https://drive.google.com/kpirgrdiagrams або відкрити фото у кращій якості з GitHub: https://github.com/blaz3xx/kpi2rgrdiagrams/tree/main

Посилання на діаграму для кращого

Опис процесу перегляду трансляції:

– користувач відкриває сторінку каналу та ініціює відтворення;

– плеєр завантажує майстер-плейлист і виконує оцінку пропускної здатності мережі (Bandwidth Estimation);

– на основі вимірювань алгоритм обирає оптимальний профіль якості (1080p, 720p або 480p);

– відбувається завантаження відповідного відеосегмента; якщо виникає помилка мережі, система повторює оцінку швидкості та пробує завантажити сегмент нижчої якості;

– успішно завантажений сегмент додається у буфер браузера (SourceBuffer) для відтворення;

– процес зациклюється: перед завантаженням кожного наступного сегмента система перевіряє, чи не завершився стрім, та заново оцінює швидкість інтернету, адаптуючи якість у реальному часі.

Процес 4 – Оформлення Premium-підписки.

Цей бізнес-процес відповідає за монетизацію платформи. Він описує взаємодію користувача з платіжним інтерфейсом, обробку транзакції на сервері та автоматичне надання розширених прав доступу.

Для опису цього бізнес-процесу використовується BPMN 2.0 модель (рисунок 1.7).

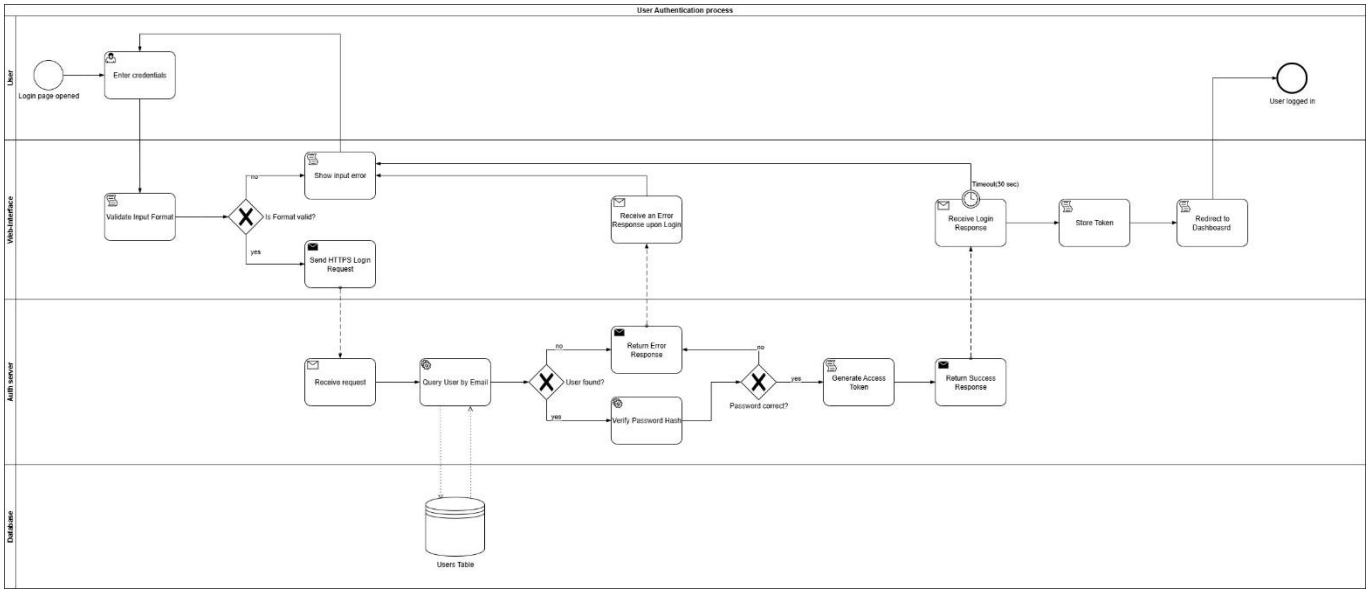

Рисунок 1.7 – Схема бізнес-процесу оформлення преміум підписки

Посилання на діаграму в середовищі draw.io для кращого перегляду(4 діаграми послідовно розташовані в 1 файлі, їх потрібно відкрити за допомогою draw.io): https://drive.google.com/kpirgrdiagrams або відкрити фото у кращій якості з GitHub: https://github.com/blaz3xx/kpi2rgrdiagrams/tree/main

Опис процесу оформлення підписки:

– користувач обирає тарифний план та вводить платіжні реквізити;

– інтерфейс виконує валідацію введених даних: у разі помилки користувачеві пропонується виправити реквізити;

– валідний запит передається на білінг-сервіс, який формує запис про інвойс та звертається до API зовнішнього платіжного шлюзу;

– у разі успішної транзакції система розраховує дату закінчення дії підписки;

– відбувається оновлення ролі користувача в базі даних (зміна статусу на Premium) та відправка електронного листа з квитанцією;

– інтерфейс відображає сторінку успішної оплати

– користувачу надходить лист на пошту про успішне придбання преміум підписки та процес активації послуги завершується

Висновки до розділу

Було проведено аналіз предметної області та виявлено низку проблем, з якими стикаються користувачі та розробники стрімінгових платформ. Головною проблемою є забезпечення стабільності відтворення відеопотоку в умовах нестабільного інтернет-з'єднання, що призводить до буферизації та затримок (latency) [5]. Також встановлено, що існуючі комерційні рішення часто накладають обмеження на авторів контенту, містять велику кількість реклами або мають закриту архітектуру, що ускладнює їх кастомізацію. Натомість, застосування сучасних протоколів сегментованої передачі даних дозволяє адаптувати якість відео під можливості мережі кінцевого користувача.

Порівнявши функціонал, визначений технічним завданням розробки, із провідними аналогами: Twitch та YouTube, було визначено, що запропонована розробка має унікальні особливості: повний контроль над затримкою трансляції, відсутність сторонньої монетизації та модульна архітектура. Розробка є актуальною, оскільки дозволяє розгортати власні незалежні платформи для освітніх чи корпоративних трансляцій, забезпечуючи при цьому високу сумісність із різними клієнтськими пристроями завдяки використанню веб-стандартів [6].

На основі проведеного аналізу алгоритмічного забезпечення було обрано Buffer-based алгоритм адаптивного бітрейту (ABR) для клієнтського плеєра, що забезпечує баланс між якістю зображення та безперервністю відтворення [3]. Для

задач доставки контенту обрано протокол HLS (HTTP Live Streaming), який гарантує надійність передачі даних через стандартні HTTP-сервери [4], а для стиснення відеопотоку — стандарт H.264, що підтримується більшістю апаратних декодерів [2].

Було визначено основні бізнес-процеси системи: авторизація користувача, обробка та трансляція потокового відео (пайплайн), адаптивне відтворення контенту та оформлення підписки. До них побудовано детальні BPMN діаграми та наведено детальний текстовий опис послідовності дій, що враховує як успішні сценарії, так і обробку помилок.

Результатом проектування є модель веб-сервісу, який дозволяє користувачеві переглядати онлайн трансляції без затримок та зависань. Система підтримуватиме автоматичну адаптацію якості відео, циклічну обробку сегментів на сервері та безпечну авторизацію, забезпечуючи при цьому високу доступність сервісу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Grigorik I. High Performance Browser Networking. Sebastopol : O'Reilly Media, 2013. С. 245–260.
2. Richardson I. E. The H.264 Advanced Video Compression Standard. 2nd ed. Chichester : Wiley, 2010. С. 112–145.
3. Li Z., Drew M. S., Liu J. Fundamentals of Multimedia. 2nd ed. Cham : Springer, 2014. С. 58–72.
4. Pantos R. HTTP Live Streaming : RFC 8216. Internet Engineering Task Force (IETF), 2017. URL: https://tools.ietf.org/html/rfc8216 (дата звернення: 01.12.2025).
5. Tanenbaum A. S., Wetherall D. J. Computer Networks. 5th ed. Boston : Pearson, 2011. С. 310–325.
6. How Video Streaming Works. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/Media (дата звернення: 01.12.2025).