

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра програмних засобів

(повне найменування кафедри )

**Пояснювальна записка**

до дипломного проекту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ

ЗАСОБІВ ВІДЕОТРАНСЛЯЦІЙ

RESEARCH AND SOFTWARE IMPLEMENTATION OF

VIDEO BROADCASTING TOOLS

Виконав: студент(ка) 2 курсу, групи КНТ-111м

Спеціальності 121 Інженерія програмного  
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Шершень А.М.

(прізвище та ініціали)

Керівник Гофман Є.О.

(прізвище та ініціали)

Рецензент Коцур М.І.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»  
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ  
Кафедра програмних засобів  
Ступінь вищої освіти магістр  
Спеціальність 121 Інженерія програмного забезпечення  
(код і найменування)  
Освітня програма (спеціалізація) Інженерія програмного забезпечення  
(назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**

Завідувач кафедри ПЗ, д.т.н, проф.  
С.О. Субботін  
“      ”                                  2022 року

**З А В Д А Н Н Я**

**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)**

Шершеня Андрія Миколайовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація засобів відеотрансляцій. Research and Software Implementation of Video Broadcasting Tools

керівник проєкту (роботи) Гофман Євгеній Олександрович, к.т.н.,  
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “04” листопада 2022 року № 363

2. Строк подання студентом проєкту (роботи) 12 грудня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

## 6. Консультанти розділів проєкту (роботи)

| Розділ              | Прізвище, ініціали та посада консультанта | Підпис, дата   |                           |
|---------------------|-------------------------------------------|----------------|---------------------------|
|                     |                                           | завдання видав | прийняв виконане завдання |
| 1-4 Основна частина | Гофман Є.О., доцент                       |                |                           |
| Нормоконтроль       | Липовець М.В., асистент                   |                |                           |

7. Дата видачі завдання “05” вересня 2022 року.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                             | Строк виконання етапів проєкту (роботи) | Примітка     |
|-------|----------------------------------------------------------------------|-----------------------------------------|--------------|
| 1     | Постановка завдання роботи.                                          | 1 тиждень                               | Завдання, ТЗ |
| 2     | Аналіз предметної області.                                           | 2-3 тижні                               | Розділ 1     |
| 3     | Вибір мови програмування та інших технологій розробки.               | 4-5 тижні                               | Розділ 2     |
| 4     | Розробка архітектури програми.                                       | 6 тиждень                               | Розділ 3     |
| 5     | Розробка програми.                                                   | 7-8 тижні                               | Розділи 3,4  |
| 6     | Тестування та експериментальне дослідження програмного забезпечення. | 9 тиждень                               | Розділ 4     |
| 7     | Оформлення пояснювальної записки та документів до неї.               | 10-11 тиждень                           | Додатки      |
| 8     | Нормоконтроль та рецензування.                                       | 12-13 тиждень                           |              |
| 9     | Захист роботи.                                                       | 14 тиждень                              |              |

Студент(ка)

\_\_\_\_\_  
( підпис )      Шершень А.М.  
(прізвище та ініціали)

Керівник проєкту (роботи)

\_\_\_\_\_  
( підпис )      Гофман Є.О.  
(прізвище та ініціали)

## РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:  
107 с., 45 рис., 5 табл., 4 дод., 17 джерел.

ANDROID DEVELOPMENT TOOLS, NODE.JS, ВІДЕОТРАНСЛЯЦІЯ,  
МОБІЛЬНИЙ ЗАСТОСУНОК, МОВА ПРОГРАМУВАННЯ, ПРОГРАМНЕ  
ЗАБЕЗПЕЧЕННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для підтримки відеотрансляцій.

Предмет дослідження – програмні засоби для відеотрансляцій.

Метою роботи є дослідження та розробка програмного забезпечення для підтримки відеотрансляцій.

Матеріали, методи та технічні засоби: плагін Android Development Tools, фреймворк Node.js, мова програмування JavaScript.

Результати. Розроблено програмне забезпечення для підтримки відеотрансляцій, яке дозволяє зареєстрованим користувачам створювати власні відеотрансляції, анонсувати їх, переглядати відеотрансляції інших користувачів.

Практична цінність роботи полягає у розробці програмного забезпечення для підтримки відеотрансляцій.

Висновки. Виконано проєктування програмного забезпечення для підтримки відеотрансляцій. Розроблено програмне забезпечення для підтримки відеотрансляцій. Здійснено тестування розробленого програмного забезпечення для підтримки відеотрансляцій.

Галузь використання – вебсервіси з підтримкою процесу спілкування між користувачами.

## **ABSTRACT**

Explanatory note to the diploma qualifying work of the master: 107 pages, 42 figures, 5 tables, 4 appendixes, 17 sources.

ANDROID DEVELOPMENT TOOLS, NODE.JS, VIDEO STREAMING, MOBILE APPLICATION, PROGRAMMING LANGUAGE, SOFTWARE.

The object of research is the process of software development for video broadcasts supporting.

The subject of the research is software for video broadcasting.

The purpose of this work is to research and develop software for video broadcasts supporting.

Materials, methods and technical tools: Android Development Tools plugin, Node.js framework, JavaScript programming language.

Results. The software tools for video broadcasts supporting that allows registered users to create their own video broadcasts, announce them, and view other users' video broadcasts have been developed.

The practical value of the work lies in the development of software for video broadcasts supporting.

Conclusions. The software for video broadcasts supporting has been designed. The software for video broadcasts supporting has been developed. The developed software for video broadcasts supporting has been tested.

Scope of use – web services that support the process of communication between users.

## ЗМІСТ

|                                                                                                                     | С. |
|---------------------------------------------------------------------------------------------------------------------|----|
| Перелік скорочень та умовних позначок.....                                                                          | 8  |
| Вступ .....                                                                                                         | 9  |
| 1 Аналіз предметної області .....                                                                                   | 10 |
| 1.1 Поняття про відеотрансляції.....                                                                                | 10 |
| 1.2 Дослідження програмних засобів для відеотрансляцій .....                                                        | 21 |
| 1.3 Висновки за розділом 1.....                                                                                     | 35 |
| 2 Матеріали і методи.....                                                                                           | 37 |
| 2.1 Вибір мови програмування та інших технологій розробки.....                                                      | 37 |
| 2.2 Вибір фреймворку для створення програмного забезпечення .....                                                   | 39 |
| 2.3 Розробка моделі взаємодії користувача з програмним забезпеченням<br>підтримки відеотрансляцій.....              | 42 |
| 2.4 Висновки за розділом 2.....                                                                                     | 47 |
| 3 Опис програми.....                                                                                                | 48 |
| 3.1 Структура програмного забезпечення для підтримки відеотрансляцій..                                              | 48 |
| 3.2 Функціонування програмного забезпечення для підтримки<br>відеотрансляцій .....                                  | 50 |
| 3.3 Проектування інтерфейсу взаємодії користувача з програмним<br>забезпеченням для підтримки відеотрансляцій ..... | 53 |
| 3.4 Опис розроблених модулів програмного забезпечення для підтримки<br>відеотрансляцій .....                        | 56 |
| 3.5 Висновки за розділом 3.....                                                                                     | 60 |
| 4 Експлуатація, тестування та експериментальне дослідження програми .....                                           | 61 |
| 4.1 Призначення й умови застосування програми.....                                                                  | 61 |
| 4.2 Характеристики програми.....                                                                                    | 62 |
| 4.3 Розгортання вебсерверу для функціонування програмного забезпечення<br>для підтримки відеотрансляцій.....        | 62 |
| 4.4 Опис роботи з програмою та результати тестування програмного<br>забезпечення для підтримки відеотрансляцій..... | 65 |

|                                    |    |
|------------------------------------|----|
| 4.5 Висновки за розділом 4.....    | 72 |
| Висновки.....                      | 73 |
| Перелік джерел посилання.....      | 76 |
| Додаток А Технічне завдання.....   | 78 |
| Додаток Б Опис програми.....       | 82 |
| Додаток В Текст програми.....      | 86 |
| Додаток Д Слайди презентації ..... | 99 |

## ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

|      |   |                                                                  |
|------|---|------------------------------------------------------------------|
| ADT  | – | Android Development Tools – засоби для розробки під Android;     |
| IP   | – | Internet Protocol – Інтернет-протокол;                           |
| OTT  | – | Over the Top – технологія передачі відеоданих;                   |
| TCP  | – | Transmission Control Protocol – Протокол керування передаванням; |
| UDP  | – | User Datagram Protocol – Протокол датаграм користувача;          |
| БД   | – | база даних;                                                      |
| ІС   | – | інформаційна система;                                            |
| ПЗ   | – | програмне забезпечення;                                          |
| СКБД | – | система керування базами даних.                                  |

## ВСТУП

Останнім часом активного розвитку набули сервіси для підтримки процесу відеотрансляцій користувачів [1]-[5].

Відомі вебсервіси забезпечення відеотрансляцій, як правило, підтримують передплатний стрімінг. В існуючих вебсервісах підтримки процесу відеотрансляцій завантаження відеотрансляцій здійснюється певними особами (адміністратори, модератори та ін.). Також часто в подібних програмах немає можливості збереження улюблених трансляцій, каналів трансляцій, а також можливості коментування відеотрансляцій. Це дещо ускладнює роботу відомих програмних засобів підтримки відеотрансляцій та обумовлює доцільність розроблення програмного забезпечення для підтримки відеотрансляцій для широкого кола користувачів.

У дипломній кваліфікаційній роботі магістра розв'язується актуальне завдання дослідження та розробки програмного забезпечення для підтримки відеотрансляцій.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для підтримки відеотрансляцій;
- розробити модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій;
- виконати проєктування програмного забезпечення підтримки відеотрансляцій;
- створити програмне забезпечення для підтримки відеотрансляцій;
- виконати тестування розробленого програмного забезпечення для підтримки відеотрансляцій.

## 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

### 1.1 Поняття про відеотрансляції

Популярність прямих відеотрансляцій постійно зростає, оскільки онлайн-відеотрансляції є одним із найкращих способів привернути увагу аудиторії. Останнім часом відео є медіаформою номер один у стратегіях контент-маркетингу, перевершуючи публікації в блогах та інфографіку. Оскільки відео зростає, і це, по суті, майбутнє бізнесу, ви можете задатися питанням, як опублікувати відео в Інтернеті [1]–[6].

Онлайн-відеомовлення — це розповсюдження відео- та/або аудіоконтенту серед аудиторії через Інтернет. Пряма трансляція — це трансляція онлайн-відео в Інтернеті в реальному часі [1]–[6].

Потокове відео в прямому ефірі швидко зростає. Зокрема, малий і середній бізнес може отримати велику користь від прямої трансляції. Компанії, які використовують відео, отримують прибуток на 49% швидше, ніж ті, які цього не роблять. Понад 55% організацій зараз використовують відео в прямому ефірі для трансляцій на всю компанію чи підрозділ. У зв'язку з цим, за оцінками експертів, до 2023 року ринок потокового відео досягне оцінки в 102,09 мільярда доларів [1]–[6].

У 2021 році через пандемію COVID-19 ми спостерігали постійне зростання онлайн-публікацій. З пом'якшенням карантину та обмежень у 2022 році компанії та організації продовжували використовувати зв'язок, який забезпечує потокове відео в прямому ефірі, щоб спілкуватися зі співробітниками, клієнтами та новими потенційними клієнтами [1]–[6].

Потокове передавання — це безперервна передача аудіо- чи відеофайлів із сервера на клієнт. Простіше кажучи, потокове передавання — це те, що відбувається, коли споживачі дивляться телевізор або слухають подкасти на пристроях, підключених до Інтернету. За допомогою потокової передачі мультимедійний файл, який відтворюється на клієнтському пристрої, зберігається віддалено та передається через Інтернет за кілька секунд за

раз [1]–[6]. Поточковий відеоконтент може включати фільми, телешоу, відео YouTube і вміст, що транслюється в прямому ефірі. Сервіси, такі як Netflix і Hulu, досягли великого успіху в залученні передплатників до поточкового відео. Термін потокове передавання означає безперервну передачу аудіо- та відеофайлів від сервера до клієнта. У відеопотоках вміст надсилається через Інтернет у стислому вигляді та переглядається глядачем у реальному часі. Медіафайли надсилаються в безперервному потоці даних і відтворюються по мірі надходження. Користувачеві потрібен плеєр, який є спеціальною програмою, яка розпаковує та передає дані зображення на екран, а аудіодані – на динаміки. Приклади медіаплеєрів включають Windows Media Player 12 для Windows 10 або QuickTime Player для macOS [1]–[6].

Потокове відеомовлення відбувається в режимі реального часу та ефективніше, ніж завантаження медіафайлів. Якщо завантажуються відеофайл, копія всього файлу зберігається на жорсткому диску пристрою, і відео не можна відтворити, доки не завершиться завантаження всього файлу. Якщо натомість транслюється, браузер відтворює відео, фактично не копіюючи та не зберігаючи його. Відео завантажуються поетапно, а не весь файл одразу, а інформація, яку завантажує браузер, не зберігається локально [1]–[6].

Завантажений відеофайл займає багато місця на жорсткому диску та потребує багато часу для переміщення. Відеопотік більше схожий на потік, оскільки відеодані безперервно та швидко надходять до браузера користувача.

Як і інші дані, що відеотрансляції надсилаються через Інтернет, аудіо- та відеодані розбиваються на пакети даних. Кожен пакет містить невеликий фрагмент файлу, а аудіо- чи відеопрогравач у браузері на клієнтському пристрої передає пакети даних і інтерпретує їх як відео чи аудіо [1]–[6].

Деякі методи поточкового передавання відеотрансляції використовують протоколи UDP, а інші — TCP. UDP і TCP є транспортними протоколами, тобто вони використовуються для переміщення пакетів даних між мережами. Обидва використовуються з Інтернет-протоколом (IP). Перед передачею даних TCP відкриває приватне з'єднання та гарантує, що всі пакети даних надходять

у порядку. На відміну від TCP, UDP не виконує нічого з цього. У результаті TCP є більш надійним, хоча передача даних через UDP займає менше часу, ніж через TCP, хоча деякі пакети втрачаються під час цього. Якщо TCP схожий на службу доставки пакетів, яка вимагає від одержувача підпису на пакеті, то UDP схожий на службу доставки, яка залишає пакети на ганку для підпису без стуку. Служба доставки TCP втрачає менше пакетів, але служба доставки UDP швидша, оскільки пакети можуть втрачати, навіть якщо вдома немає нікого, хто б їх підписав [1]–[6].

У деяких випадках для потокового передавання швидкість важливіша за надійність. Наприклад, якщо хтось бере участь у відеоконференції, він краще буде спілкуватися в режимі реального часу з іншими учасниками конференції, ніж сидіти й чекати кожного біта даних. Тому кілька втрачених пакетів даних не є серйозною проблемою, тому слід використовувати UDP [1]–[6].

В інших випадках для потокової передачі важливіше надійність. Наприклад, потокове передавання HTTP (HLS) і MPEG-DASH є потоковими протоколами, які використовують TCP для передачі. Багато служб відео на вимогу використовують TCP [1]–[6].

Потокові медіапрогравачі попередньо завантажують потік на кілька секунд, щоб відео чи аудіо могли продовжувати відтворення, якщо з'єднання ненадовго перервано. Це відомо як буферизація. Буферизація забезпечує плавне та безперервне відтворення відео. Однак, якщо є велика затримка через повільне з'єднання або в мережі, буферизація відео може тривати довго [1]–[6]. Відеопотоки зазвичай надсилаються з попередньо записаного відеофайлу, але їх також можна поширювати як частину прямого ефіру. Під час прямої трансляції відеосигнал перетворюється на стислий цифровий сигнал і передається через спеціальний веб-сервер, який або здійснює групову передачу, або надсилає той самий файл кільком користувачам одночасно [1]–[6].

Користувачі можуть транслювати зі своїх настільних комп'ютерів, смартфонів, планшетів і смарт-телевізорів або інших пристроїв, таких як Chromecast або Apple TV. Для трансляції відео можна використовувати різні програми, наприклад YouTube, Netflix, Amazon Prime Video, Disney+ або Twitch. Такі сервіси, як YouTube, транслюють відео безкоштовно та підтримуються рекламою, а такі сервіси, як Netflix, працюють на основі моделі підписки користувача [1]–[6].

Потокове відео також вимагає досить високої швидкості для оптимальної продуктивності. Нижча якість потокового відео вимагає менше даних, але вищої якості відео, як-от 1080p із прогресивною розгорткою, або вищі швидкості передачі даних для плавного відтворення 4K [1]–[6].

У той час як невелика кількість телевізійних шоу транслюється до 1080i за допомогою плагіна екрана, більшість шоу транслюється бездротовим способом із стандартом основної роздільної здатності 720p, роздільною здатністю, яку YouTube більше не вважає високою чіткістю. Для порівняння, потокові сервіси, такі як Netflix і YouTube, пропонують роздільну здатність відео до 2160p, тобто 4K [1]–[6].

Відеотрансляції у вигляді потокового відео забезпечують користувачів такими перевагами [1]–[6]:

- можливість завантаження та трансляції. Користувачі можуть переглядати відео без необхідності їх завантажувати [1]–[6];

- висока роздільна здатність відтворення. Деякі служби підтримують роздільну здатність до 4K, яка є вищою за стандарт роздільної здатності бездротового телебачення [1]–[6];

- ціна. Хоча деякі сервіси потокового передавання відео, як-от YouTube і Twitch, доступні безкоштовно, інші сервіси потокового передавання покладаються на моделі підписки, які можуть коштувати дешевше, ніж середня кабельна підписка [1]–[6];

- можливість вибору платформи. Користувачі можуть вибирати з кількох платформ для потокового відео. Сервіси потокового телебачення та

фільмів часто намагаються конкурувати один з одним, створюючи ексклюзивний контент. Користувачі можуть у будь-який час переглядати контент, що транслюється в прямому ефірі, одночасно переглядаючи вміст за запитом, який є попередньо записаним медіафайлом [1]–[6];

– варіація змісту. Оскільки концепція потокового відео настільки широка, люди можуть транслювати телешоу, фільми, контент, створений користувачами, із таких вебсайтів, як YouTube, або в режимі онлайн [1]–[6].

Однак застосування концепції потокового відеоконтенту пов'язано з певними проблемами. До них належать [1]–[6]:

– відсутність пропускної здатності. Що вища якість відео, то більша пропускна здатність потрібна для підтримки потокового передавання даних. Якщо кілька пристроїв використовують великі обсяги даних одночасно, пропускної здатності може бути недостатньо для потокової передачі високого вмісту в 4К або іншій високій якості [1]–[6];

– повільна робота клієнтських пристроїв. Якщо пристрій старий, уповільнюється через те, що програми не оптимізовані для нього, або на ньому одночасно запущено занадто багато процесів, відеопотоки також можуть сповільнюватися або втрачати швидкість;

– авторське право. Користувачі потокового відеовмісту – особливо потокового вмісту в прямому ефірі – повинні знати про обмеження щодо вмісту, захищеного авторським правом, наприклад музики, яка з'являється в їхніх потоках. Це серйозна проблема на платформах створеного користувачами контенту, таких як YouTube і Twitch [1]–[6];

– затримка мережі. З боку мережі розташування клієнтів і серверів може впливати на затримку. Чим далі клієнт і сервер, тим більшу відстань повинні пройти дані [1]–[6].

Тенденції та статистика прямих трансляцій [1]–[6]:

– очікується, що вартість глобальної галузі онлайн-відеоплатформ зросте на 8,4% з 2021 по 2028 рік;

- за перші 6 місяців пандемії кількість подій у прямому ефірі зросла на 300%;
- 87% відеомаркетологів повідомляють, що використання живого відеоконтенту збільшує відвідуваність їх сайту;
- очікується, що до 2027 року вартість ефірної індустрії становитиме 247 мільярдів доларів;
- живий контент утримує увагу глядачів у 10-20 разів довше, ніж відео на замовлення;
- сайти з відео мають у 50 разів більше шансів отримати рейтинг у лише текстових результатах пошуку;
- у 2022 році багатопотокове передавання стане великим, і багато багатопотокових платформ уже доступні для потокового передавання на кілька джерел.

Статистика показує, що прямі трансляції в Інтернеті є сильними та постійними [1]–[6].

Потокове відео в прямому ефірі може бути корисним для багатьох речей, наприклад для спортивних ігор, церковних служб або запуску продуктів. Пряма трансляція таких подій дозволяє людям бути присутніми, де б вони не були [1]–[6].

Більшість онлайн-потоків підтримується технологією потокового передавання Over-the-Top (OTT). Наразі понад 64 мільйони домогосподарств у США використовують поточкові послуги OTT на додаток до або замість традиційних методів доставки відео, таких як супутникове чи кабельне телебачення. Потокове відео стало звичним явищем у домівках США [1]–[6].

У 2018 році світовий дохід від OTT склав приблизно 68 мільярдів доларів. Це на 33% більше, ніж у 2017 році. Експерти прогнозують, що до 2026 року ця цифра зросте до 210 мільярдів доларів [1]–[6].

Розміщення онлайн-потоків відео вимагає певної підготовки. Процес налаштування трансляції виглядає дещо по-різному для кожного стримера, оскільки це залежить від ваших унікальних цілей і потреб [1]–[6].

Розглянемо деякі загальні кроки для підготовки професійного налаштування потокового передавання. Ці загальні кроки описують процес трансляції відео в прямому ефірі [1]–[6].

На першому етапі розробляється план. Кожен успішний ефір починається з твердого плану. Перш ніж почати створювати налаштування потокового передавання, знайдіть час, щоб визначити, чого ви хочете досягти за допомогою свого відеоканалу. Складіть список ваших конкретних цілей і які функції вам знадобляться для досягнення цих цілей [1]–[6].

Необхідно зрозуміти, кого ви хочете охопити своїми потоками та чому ви хочете їх охопити. Також важливо отримати оцінку того, скільки людей переглядатимуть ваші потоки, оскільки це допоможе визначити, яка пропускна здатність вам знадобиться.

Знання того, чим ви хочете поділитися та кого ви хочете охопити своїм вмістом, допоможе вам вибрати найкраще налаштування прямої трансляції.

На наступному етапі необхідно обрати обладнання та, відповідно, інвестувати у камеру та аудіообладнання. Для трансляції потрібні різні види обладнання [1]–[6].

Перше обладнання, яке знадобиться бізнесу для трансляції відео, — це камера для трансляції в прямому ефірі. Доступні тисячі різних камер для трансляції в прямому ефірі, але не перевантажуйте їх можливістю. Як і більшість продуктів, відеокамери для потокового відео в прямому ефірі мають компроміси [1]–[6].

Відеокамери початкового рівня, як правило, дешевші та легші. Якість відео може бути хорошою, але лише за ідеальних умов освітлення. Вони не мають ручного керування та мають лише базові можливості підключення. Ці камери коштують від 300 до 800 доларів США, включаючи камери смартфонів, екшн-камери та камери початкового рівня [1]–[6].

Професійні відеокамери оснащені професійними роз'ємами, такими як XLR і SDI. Вони, як правило, мають великі датчики з високою роздільною здатністю, які забезпечують чудову якість зображення навіть в умовах

поганого освітлення. З ручним керуванням якість збірки вища, що робить їх довговічнішими. Деякі професійні камери навіть дозволяють використовувати змінні об'єктиви. Ці камери коштують від 1500 до 25 000 доларів і включають професійні камери, камери EFT і кінокамери [1]–[6].

Якщо у вас є відеокамера, аудіо є наступним важливим елементом обладнання для потокового відео [1]–[6].

Багато камер також мають якісний вбудований мікрофон. Однак ви можете придбати зовнішній мікрофон для високоякісного звуку. Це також дозволить адаптуватися до поточних умов. Обладнання для аудіотрансляції в прямому ефірі має три основні рівні [1]–[6]:

- обладнання середньо-низької якості: вбудований мікрофон у початкових і професійних споживчих камерах;
- обладнання середньо-високої якості: мікрофони USB і 3,5 мм або вбудовані мікрофони на професійних відеокамерах;
- обладнання професійної якості: мікрофони XLR.

Можна використовувати накладний мікрофон для інтерв'ю або спрямований мікрофон, щоб усунути фоновий шум [1]–[6].

Базовий дротовий петличний мікрофон 3,5 мм за 16 доларів може значно покращити якість звуку порівняно з відеокамерами початкового рівня чи вбудованими мікрофонами в телефонах. Ви також можете оновити мікрофон високого класу 3,5 мм для кращої якості. Якщо ви хочете максимально підвищити якість звуку, розгляньте мікрофони XLR. Це професійний стандарт якості та довговічності, який існує десятиліттями.

Далі необхідно обрати програмних кодер (кодувальник). Відеокодер — це інструмент, який перетворює ваш відеопотік із файлу RAW у цифровий і готує його до трансляції. Це дає змогу передавати відео від джерела до глядача через Інтернет [1]–[6].

Існує два основних типи кодерів: апаратні кодери та програмні кодери.

Апаратні кодери — це спеціалізовані процесори, які використовують алгоритм, призначений для кодування відео та даних у потоковий вміст. Ці

кодери поставляються в менших портативних коробках або більших стаціонарних пристроях. Вони підтримують широкий спектр джерел, але найчастіше HDMI і SDI [1]–[6].

Апаратні кодери, як правило, створені для тривалого використання та можуть підтримувати цілодобову потокову передачу відео без збоїв, що робить їх ідеальними для складних випадків потокового використання. На ринку також є кілька спеціалізованих апаратних кодерів, які використовують живлення акумулятора та стільникові з'єднання для передачі відео в реальному часі з віддалених місць [1]–[6].

З іншого боку, програмне забезпечення для кодування прямих трансляцій працює на обчислювальному пристрої, такому як ноутбук або настільний комп'ютер. Щоб знімати відео, їх потрібно об'єднати в пару з платою захоплення або іншим способом під'єднати камери до комп'ютера. Ці програмні пакети також інтегрують інструменти змішування та виробництва.

Програмні кодери привабливі для початківців і проміжних видавців через їх низьку вартість і функції налаштування. Але програмним кодувальникам не вистачає загальної затримки, яку пропонують апаратні кодувальники, і це потрібно враховувати, вибираючи правильний кодер для своєї прямої трансляції [1]–[6].

Як і у випадку з камерами, ви можете знайти програмне забезпечення для кодування в широкому діапазоні цін: від безкоштовного до дорогого професійного рівня. Найкращий вибір залежить від того, яку трансляцію ви хочете зібрати, скільки джерел відеоканалу ви використовуєте та скільки модної роботи з відео ви хочете додати [1]–[6].

На четвертому етапі необхідно забезпечити надійне підключення до Інтернету. Якими б не були ваші потокові цілі, надійне підключення до Інтернету є обов'язковим. Вам знадобиться надійне підключення до Інтернету з достатньою швидкістю завантаження. Як правило, вам потрібно, щоб швидкість завантаження була вдвічі швидшою за бітрейт, з яким ви транслюєте [1]–[6].

Для надійної прямої трансляції у форматі Full HD зі швидкістю 10 Мбіт/с ми рекомендуємо швидкість завантаження приблизно 20 Мбіт/с. Але якщо ви транслюєте дуже бітрейт, вам знадобиться вища швидкість завантаження [1]–[6].

У більшості налаштувань потокового передавання бажано підключення до Інтернету через прив'язку, але може бути достатньо Wi-Fi або стільникового зв'язку. Для віддаленої трансляції найкращим варіантом є використання смартфона з мобільною точкою доступу. Якщо вам потрібна більша пропускна здатність, ви можете скористатися стільниковим рішенням.

На п'ятому етапі необхідно обрати платформу онлайн-відео [1]–[6].

Платформа онлайн-відео — це програмне забезпечення, яке розміщує ваше відео, доставляє його глядачам і дозволяє легко вставляти живі відео на ваш вебсайт. Це дає вам можливість транслювати відео в прямому ефірі для аудиторії, а потім записувати цей потік на випадок, якщо інші знайдуть його та захочуть переглянути його колись [1]–[6].

Для бізнесу є багато варіантів. Найкращим варіантом є професійне відеорішення білої мітки, таке як Dacast. Dacast підтримує як живе відео, так і потоковий вміст VOD. Це «послуга з білою міткою», оскільки ваша аудиторія не бачить логотип або ідентифікатор служби. Ви можете легко додати свій логотип, щоб створити враження від перегляду, що відповідає бренду. Для професійного процесу публікації відео потрібна онлайн-платформа відео [1]–[6].

Шостий етап полягає у встановленні налаштувань, після чого можна розпочати трансляцію. Необхідно підключити джерела до онлайн-відеопрогравача, налаштувати параметри кодера та вбудувати відеоплеєр [1]–[6].

Фактори, які можуть призвести до уповільнення відеотрансляцій на стороні сервера такі [1]–[6]:

- латентність (затримка) мережі. На затримку впливають різні фактори, зокрема те, де зберігається вміст, до якого користувачі намагаються отримати доступ [1]–[6];

- перевантаження мережі: якщо через мережу надсилається забагато даних, це може погіршити продуктивність потокового передавання [1]–[6].

Фактори, які можуть призвести до уповільнення відеотрансляцій з боку користувача такі [1]–[6]:

- проблеми з Wi-Fi: перезапуск маршрутизатора локальної мережі або перемикання на Ethernet замість WiFi може допомогти покращити продуктивність потокового передавання [1]–[6];

- повільні клієнтські пристрої: відтворення відео вимагає великої потужності процесора. На продуктивність потокової передачі може вплинути, якщо на пристрої, що передає відео, запущено багато інших процесів або він загалом повільний [1]–[6];

- недостатня пропускна здатність: домашні мережі потребують близько 4 Мбіт/с для потокового відео; для відео високої чіткості їм, ймовірно, знадобиться більше [1]–[6].

Отже, потокове передавання зазнає тих самих затримок і зниження продуктивності, що й інші типи веб-вмісту. Як і у випадку з будь-яким типом вмісту, доступ до якого здійснюється через Інтернет, розташування хостингу має велике значення, оскільки потоковий вміст зберігається в іншому місці. Якщо користувач із Нью-Йорка намагається транслювати з сервера Netflix у Лос-Гатос, відеоконтенту доведеться подолати 3000 миль, щоб досягти користувача, і відео доведеться довго буферизувати або може не відтворюватися. немає. Ось чому Netflix та інші провайдери потокового передавання широко використовують розподілені мережі доставки контенту, які зберігають вміст набагато ближче до користувачів у всьому світі. Розподілені мережі доставки контенту мають досить позитивний вплив на продуктивність потокового передавання [1]–[6].

Таким чином, проаналізовано поняття відеотрансляцій, досліджено основні етапи, які необхідно здійснити для організації та проведення відеотрансляцій. Відомі вебсервіси є аналогами відеотелепередач, які можуть дивитися користувачі у зручний для них час на різних апаратних платформах.

## 1.2 Дослідження програмних засобів для відеотрансляцій

Останнім часом широко використовуються програмні засоби для організації та підтримки відеотрансляцій. Виконаємо аналіз деяких з них [6]–[12].

У процесі порівняння та дослідження потокових платформ для онлайн-трансляції будемо аналізувати основний функціонал системи, її характеристики, переваги та недоліки, принципи ціноутворення щодо придбання платних версій, а також можливість використання для бізнесу.

Програмне забезпечення Brightcove (рис. 1.1) — онлайн-відеоплатформа, орієнтована на великий бізнес і заснована в 2004 році Джеремі Аллером і Бобом Мейсоном [2], [7].



Рисунок 1.1 – Програмне забезпечення Brightcove [7]

Brightcove поділяє свою продукцію на кілька різних категорій. До них належать Video Player, OTT Streaming, Video Marketing Bundle та інші. Серед інших пропозицій, платформа Brightcove для потокової передачі та хостингу називається Video Cloud. Ця платформа містить повноекранний програвач і підтримує монетизацію (рис. 1.1) [2], [7].

Основні риси програмного забезпечення Brightcove [2], [7]:

- технологія Brightcove забезпечує можливість перемогти блокувальники реклами та додати відеорекламу до відеопотоків. Хоча незрозуміло, чи може цей метод перемогти останнє покоління блокувальників реклами, він все ще цінний для компаній, які використовують цю модель монетизації вмісту;

- наявність функціональних маркетингових інструментів, що дозволяє ефективно використовувати цю платформу для розв'язання бізнес-завдань.

- можливість монетизації та реклами;
- наявність відеопрогравача HTML5 для всіх пристроїв;
- засоби відеоаналітики;
- інструменти соціальної публікації
- засоби REST API на основі маркерів;
- цілодобова підтримка клієнтів.

Перевагами програмного забезпечення Brightcove є такі [2], [7]:

- можливість керування відеотекою;
- підтримка відео HTML5;
- відеоаналітика;
- пряма трансляція події;
- варіанти монетизації.

Недоліками програмного забезпечення Brightcove є такі [2], [7]:

- не дуже зручний для користувача;
- недостатні засоби підтримки користувача;
- відсутність вбудованих імен;
- необхідність оплачувати за підтримку 24/7.

Технічні характеристики програмного забезпечення Brightcove є такі [2], [7]:

- мінімальний розмір: визначається користувачем;
- максимальний розмір: визначається користувачем;
- співвідношення сторін: без обмежень (оскільки Brightcove використовує адаптивний програвач);
- максимальний розмір файлу: немає обмежень на розмір файлу;
- максимальна довжина відео: немає обмежень для публікації;
- загальна пам'ять для файлів: залежить від плану;
- сумісні відеоформати: MP4, MOV, FLV, AVI, WMV, MKV, 3GPP.

Ціни: оскільки Brightcove в основному націлений на клієнтів корпоративного рівня, він пропонує кілька планів потокового передавання за особливими цінами за вищими цінами [2], [7].

До трьох рівнів ціноутворення входять [2], [7]:

- початковий: включає від 50 до 200 відео, до 100 000 відтворень і 2-5 користувачів;
- Pro: Необмежена кількість відео для більше, ніж десяти користувачів, більше 100 000 відтворень і без повторних відтворень;
- підприємство: необмежену кількість відео для більше, ніж десяти користувачів, необмежену кількість завантажень і лише користувачів аналітики, понад 100 000 відтворень без додаткових відтворень.

Brightcove найкраще підійде для професійних мовників із потребами корпоративного рівня в прямому ефірі. Ця платформа в першу чергу орієнтована на мовників, видавців, маркетологів, підприємства, роздрібну торгівлю та спортивну індустрію [2], [7].

Програмне забезпечення Dacast [2], [8] представляє собою платформу для трансляції та розміщення відео в Інтернеті. Dacast — це повнофункціональне рішення для потокового передавання, яке включає професійну уніфіковану платформу для потокового передавання в прямому ефірі та відео на вимогу (рис. 1.2) [2], [8].

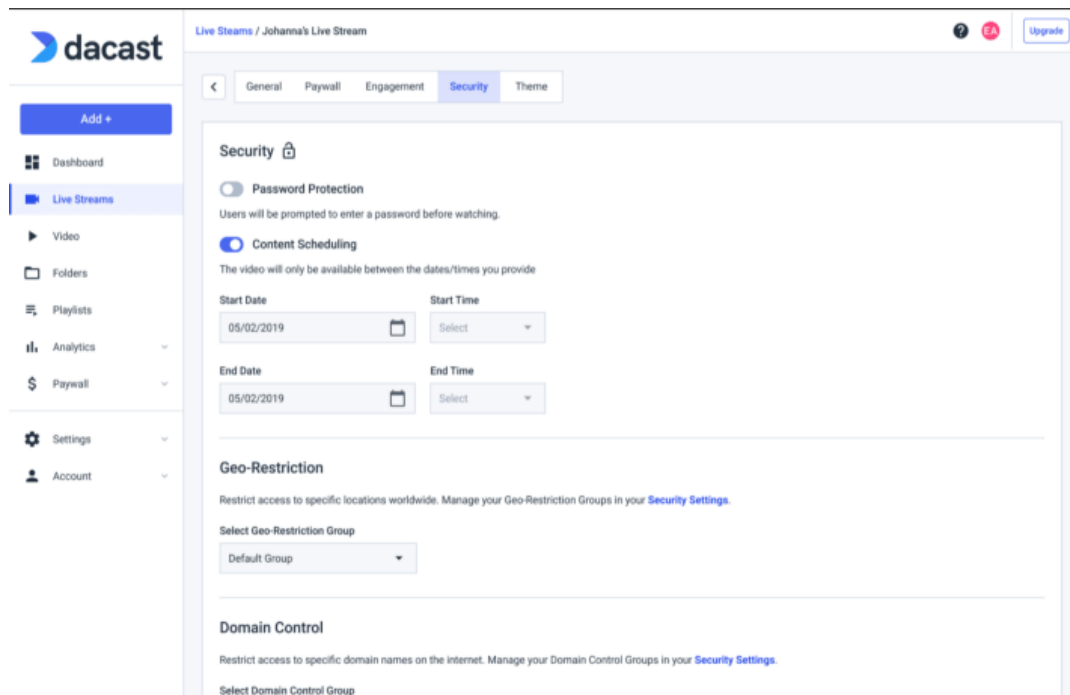


Рисунок 1.2 – Програмне забезпечення Dacast [8]

Dacast пропонує комплексні послуги щодо реалізації потокового відео в прямому ефірі та розміщення онлайн-відео. Підприємства можуть вставляти відео на власні вебсайти, зберігаючи 100% контроль над своїм вмістом. Серед інших функцій Dacast пропонує інтегровану систему оплати, захист паролем, обмеження рефералів і панель аналітики [2], [8].

Основні риси програмного забезпечення Dacast [2], [8]:

- Dacast вирізняється тим, що пропонує високоякісні функції з конкурентними ціновими планами. Це потокове передавання без реклами, безкоштовне потокове передавання на всіх планах і преміальна доставка CDN. Цілодобова підтримка по телефону також доступна для планів Activity і Scale;
- простота використання та доступність. Навіть нові мовники можуть створити новий живий канал і розпочати нову пряму трансляцію всього за кілька хвилин [2], [8].

Детальні характеристики програмного забезпечення Dacast [2], [8]:

- потокове відео в прямому ефірі;
- пряма трансляція відео;

- глобальне розповсюдження контенту, включаючи найкращий бізнес-відеохостинг у галузі;

- функції безпеки світового рівня;
- пряма трансляція, захищена паролем;
- прийом, відтворення та підтримка потокового передавання;
- багатобітрейтове потокове передавання для живого відео;
- хмарне транскодування відео зі справжньою адаптивною потоковою швидкістю потоку;

- підтримка кодування в реальному часі для найкращих відеокодерів;
- цілодобова телефонна підтримка з преміум-планами;
- необмежена кількість каналів у прямому ефірі;
- аналітика в реальному часі;
- шифрування AES для безпечного потокового відео;
- інтеграція Zoom live streaming для зустрічей і подій у реальному часі;
- відеопортал галерей Exro 2.0 для захоплюючого відео;
- потокове передавання через браузер;
- китайський варіант прямої трансляції.

Переваги програмного забезпечення Dacast такі [2], [8]:

- налаштовуваний відеопрогравач HTML5 із опціями вбудовування;
- цілодобова технічна підтримка з усіма планами, включаючи електронну пошту та чат;

- перекодування та кодування відео;
- доступ до API відео та API програвача з преміум-планами;
- китайський відеохостинг;
- трансляція без реклами.

Недоліком програмного забезпечення Dacast є те, що його використання вимагає деякого навчання [2], [8].

Технічні характеристики програмного забезпечення Dacast є такі [2], [8]:

- мінімальний розмір: немає (але рекомендований мінімум 240p);

- максимальний розмір: 1080p або 4K (залежить від апаратного забезпечення користувача та Інтернету);
- співвідношення сторін відео: без обмежень (але за замовчуванням 16:9);
- максимальний розмір відеофайлу: немає;
- максимальна довжина відео: немає;
- загальний обсяг файлів: 10-1000 ГБ (залежить від плану);
- сумісні відеоформати: MP4 (бажано), MOV, M4V, M2V, AVI, MPG, FLV, WMV, MKV, WEBM, OGV, MXF, ASF, VOB, MT.

Щомісячне обслуговування з Dacast доступне в трьох рівнях тарифного плану. Пакети планів такі [2], [8]:

- початковий пакет вартістю \$39/місяць (рахунок виставляється щорічно). Ідеально підходить для новачків із 1,2 ТБ пропускної здатності та 50 ГБ пам'яті на рік;

- середній пакет вартістю \$63/місяць (або \$750/рік) – пропонує 6 ТБ попередньої пропускної здатності, 50 ГБ пам'яті для організацій, які не хочуть підписувати контракт і хочуть окремо придбати пропускну здатність для прямої трансляції події. За потреби можна додати додаткові дані та сховище;

- масштабний пакет: \$188/місяць (рахунок виставляється щорічно); включає 1 ТБ пам'яті та 24 ТБ пропускної здатності на рік; додайте підтримку по телефону, платний екран для монетизації, онлайн-програвач M3U8 для живих каналів, вставлення реклами, відеопрогравач із білою міткою та доступ до API відео; додаткова пропускну здатність доступна за запитом для місячних планів [2], [8].

Dacast добре підходить для професійних мовників, яким потрібна уніфікована потокова платформа з розширеними можливостями хостингу та цілодобовою підтримкою клієнтів [2], [8].

Програмне забезпечення Dacast ідеально підходить для таких завдань та організацій [2], [8]:

- підтримка відеопотоків підприємств;

- дистанційне навчання;
- пряма спортивна трансляція;
- ЗМІ та агентства;
- церкви та культові споруди;
- урядові установи.

Програмне забезпечення Kaltura [2], [9] (рис. 1.3) дозволяє здійснювати відеотрансляції та потребує серверної інфраструктури для розміщення та розповсюдження. Програмне забезпечення Kaltura пропонує хмарні відеохостинги та послуги розповсюдження як для відкритих трансляцій, так і для прямих трансляцій.



Рисунок 1.3 – Програмне забезпечення Kaltura [2], [9]

Програмне забезпечення Kaltura [2], [9] є досить добре розширюваним. Природа програмного забезпечення з відкритим кодом означає, що спільнота користувачів створює та ділиться незліченною кількістю плагінів та інтеграцій.

Недоліком програмного забезпечення Kaltura [2], [9] є те, що використання платформи може заплутати, тобто наявність великої кількості функцій, які іноді є зайвими. З такою кількістю різноманітних функцій, це може бути приголомшливим для початківців.

Детальні характеристики програмного забезпечення Kaltura [2], [9]:

- наявність мультимедіа та адаптивного потокового передавання;
- глобальне розповсюдження контенту з уніфікованим підходом до кількох CDN (з можливістю залучення власного CDN);
- технічна підтримка через систему на основі квитків, яка працює повільніше, ніж чат або телефон;
- доступ до API відео;
- функції безпеки, включаючи ряд параметрів автентифікації та шифрування;
- монетизація, включаючи всі основні моделі (підписка, реклама, потокове відео PPV).

Перевагами програмного забезпечення Kaltura [2], [9] є такі:

- багато доступних функцій;
- хороші варіанти монетизації;
- сильна безпека.

Недоліками програмного забезпечення Kaltura [2], [9] є такі:

- відсутність підтримки по телефону чи чату;
- досить складний інтерфейс.

Технічні характеристики програмного забезпечення Kaltura [2], [9] є такі:

- мінімальний розмір: 180p (320 x 180);
- максимальний розмір: 1080p (1920x 1080);
- співвідношення сторін: 16:9, 4:3;
- максимальний розмір файлу: 2 Гб;
- максимальна довжина відео: недоступна;

- загальний обсяг пам'яті файлів: 10 ГБ – необмежений (залежить від плану);
- сумісні відеоформати: MOV, MP4.

Програмне забезпечення Kaltura не публікує точні ціни на своєму вебсайті, оскільки кожен план є гнучким. Натомість видавці можуть придбати кредити, щоб витратити їх на пропускну здатність і доступ до різних інтеграцій [2], [9].

Програмне забезпечення Kaltura добре підходить для демонстрації ваших відео та використання відео для навчання, навчання, спілкування, співпраці та розваг. Історично ця компанія найбільше працювала з навчальними закладами [2], [9].

Програмне забезпечення Flowplayer пропонує рішення для хостингу для потокового онлайн-відео [2], [10]. Що стосується пропозицій функцій, Flowplayer відомий білими сервісами потокового передавання та преміальними рішеннями для монетизації [2], [10]. Основна мета Flowplayer — надати малим підприємствам інструменти, необхідні для розміщення, трансляції та монетизації відео за допомогою стильного та професійного відеопрогравача [2], [10].

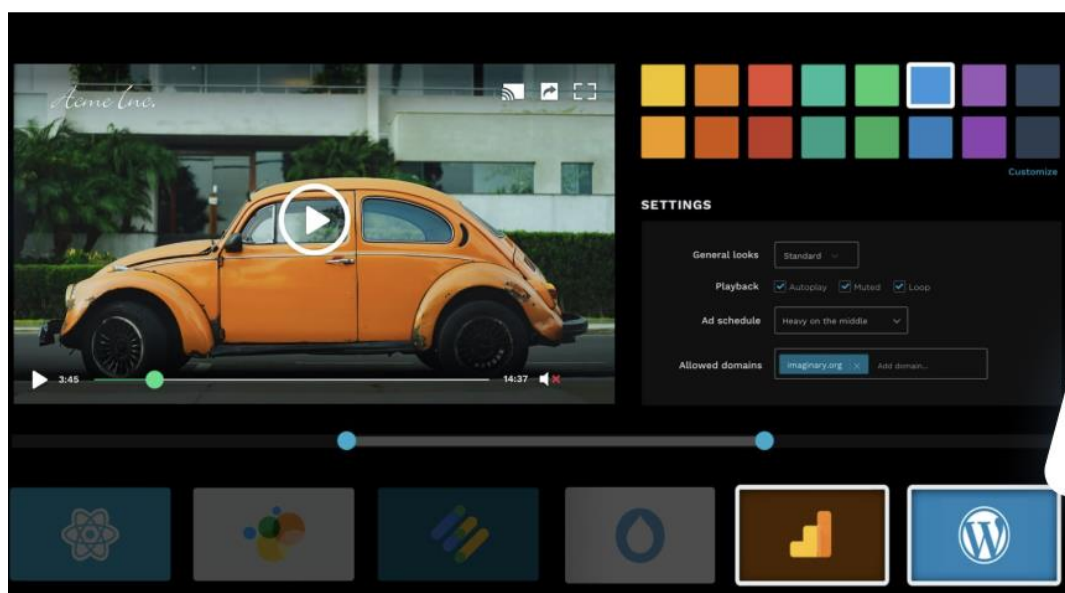


Рисунок 1.4 – Програмне забезпечення Flowplayer [2], [10]

Програмне забезпечення Flowplayer (рис. 1.4) є досить надійним відеоплеєром і інструментами монетизації. Він також має дуже чітку ціну [2], [10].

Детальні характеристики програмного забезпечення Flowplayer [2], [10]:

- високоякісний відеоплеєр;
- можливості прямих трансляцій;
- відзначені нагородами рішення для монетизації;
- потужна відеоаналітика.

Перевагами програмного забезпечення Flowplayer є такі [2], [10]:

- надійний відеоплеєр;
- стильний інтерфейс.

Технічні характеристики програмного забезпечення Flowplayer [2], [10] є такі:

- мінімальні розміри: 426 x 240;
- максимальний розмір: 3840 x 2160;
- підтримувані співвідношення сторін: 16:9 (за замовчуванням);
- максимальний розмір файлу: недоступний;
- максимальна довжина відео: недоступна;
- загальна пам'ять для файлів: 50 ГБ, 100 ГБ або індивідуальна (залежно від плану);
- прийнятні формати відео: MP4 (рекомендовано).

Програмне забезпечення Flowplayer має два тарифні плани, зокрема:

- базовий: \$49/місяць, 500 ГБ пропускної здатності, включає 50 000 відтворень щомісяця, спеціальний програвач, керування відео, списки відтворення та аналітику;

– підприємство: спеціальні ціни, створено на замовлення для задоволення розширених потреб потокового передавання.

Програмне забезпечення Flowplayer добре підходить для стримерів, які шукають універсальну платформу, щоб вивести ваше відео на новий рівень.

Ця онлайн-відеоплатформа обслуговує видавців, видавців, цифрові медіа-будинки та малий і середній бізнес.

Програмне забезпечення Wistia (рис. 1.5) призначено розміщення онлайн-відеоконтенту. Ця платформа підходить для розміщення онлайн-відео, але не пропонує функції потокового передавання [2], [11]. Програмне забезпечення відеомаркетингу Wistia має головною мету залучити та збільшити аудиторію бізнесу. Програмне забезпечення Wistia включає рішення для реєстрації, хостингу, співпраці та керування [2], [11].

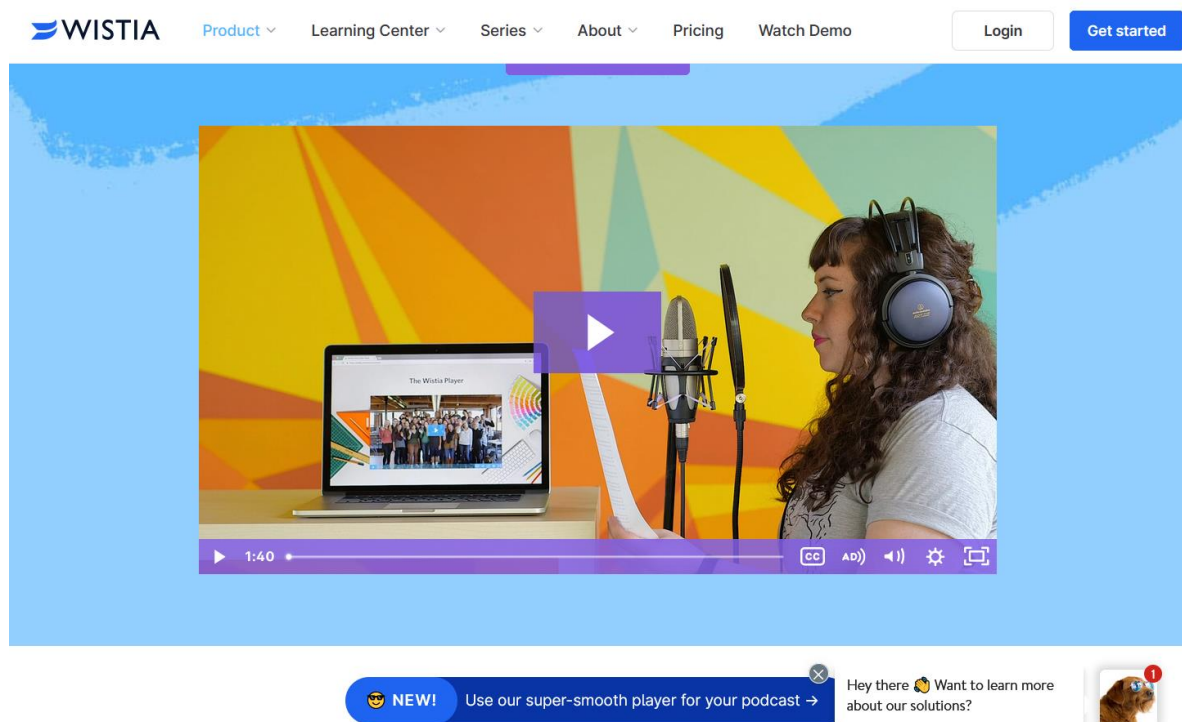


Рисунок 1.5 – Програмне забезпечення Wistia [2], [11]

Ключовою особливістю програмного забезпечення Wistia є те, що воно надає своїм глядачам якісний телевізійний досвід, який допомагає їм розширити аудиторію. Програмне забезпечення Wistia є онлайн-відеоплатформою, яка дає користувачам можливість контролювати їхній вміст [2], [11].

Детальні характеристики програмного забезпечення Wistia такі [2], [11]:

- мультибітрейт і адаптивна потокова передача;
- глобальне поширення контенту;
- технічна підтримка, але підтримка преміум-класу доступна лише за спеціальними планами;
- доступ до Video API із шістьма різними доступними API;
- функції безпеки, включаючи захист паролем, ретрансляцію електронної пошти та обмеження домену.

Перевагами програмного забезпечення Wistia є такі [2], [11]:

- зручна аналітика;
- майже необмежена пропускна здатність;
- інтеграція з різними CRM;
- функції безпеки.

Недоліками програмного забезпечення Wistia є такі [2], [11]:

- відсутність прямої трансляції;
- відсутність можливості заробляння грошей;
- додаткові відео збільшують вартість.

Технічні характеристики програмного забезпечення Wistia [2], [11] є такі:

- мінімальні розміри: немає;
- максимальні розміри: максимум 4K (3840 x 2160);
- підтримувані пропорції: будь-які;
- максимальний розмір файлу: 8 ГБ;
- максимальна тривалість відео: 2 години;
- загальне сховище файлів: до 10 відео (99 доларів США на місяць) із професійним планом, 0,25 доларів США на місяць за кожне додаткове відео;
- прийнятні формати відео: MP4 (рекомендований формат), .MOV, .AVI, .WMV, .FLV, .MKV, 3GPP.

Програмне забезпечення Wistia має три рівні тарифних планів [2], [11], зокрема:

- безкоштовний: жодних витрат на використання, до 250 підписників і 3 відео, неможливо видалити бренд Wistia з відеопрогравача;
- професійний: 99 доларів США на місяць, 10 безкоштовних відео з додатковими відео по \$0,25 за кожне, відеоплеєр white label;
- розширений: індивідуальне ціноутворення, 100 безкоштовних відео з додатковими відео по \$0,25 за кожне, кілька каналів, 250+ підписників.

Програмне забезпечення Wistia найкраще підходить для потреб відеомаркетингу. Такі компанії, як Mailchimp і Moz, використовували Wistia для своїх маркетингових можливостей [2], [11].

Wistia призначена для бізнесу, який хоче розвивати свій бренд за допомогою відео. Ця платформа пропонує відео SEO та інструменти, які допомагають створювати оголошення в пошукових і соціальних мережах для маркетингових цілей [2], [11].

Таким чином, програмні засоби для забезпечення відеотрансляцій [2]-[11] є досить гарним варіантом для компаній, які прагнуть вивести свій бізнес на новий рівень, зв'язавшись зі своєю цільовою аудиторією за допомогою цього захоплюючого типу вмісту.

Порівняльний аналіз програмного забезпечення для підтримки відеотрансляцій наведено у табл. 1.1.

Як видно з таблиці, відомі програмні засоби для підтримки відеотрансляцій, як правило, вимагають оплати для авторів відеотрансляцій, надають користувачам засоби відеоаналітики, маркетингові інструменти та дозволяють монетизувати свою діяльність. Проте у деяких наявних програмних засобах для підтримки відеотрансляцій не підтримується можливість коментування відеотрансляції, а деякі інші подібні програми є досить складними для використання на практиці, особливо людям, які не є фахівцями з відеотрансляцій. Отже доцільно розробити програмне забезпечення для підтримки відеотрансляцій.

Таблиця 1.1 – Програмне забезпечення для підтримки відеотрансляцій

| Критерій, за яким порівнюють                 | Brightcove [2], [7] | Dacast [2], [8] | Kaltura [2], [9] | Flowplayer [2], [10] | Wistia [2], [11] |
|----------------------------------------------|---------------------|-----------------|------------------|----------------------|------------------|
| Наявність засобів відеоаналітики             | +                   | +               | +                | —                    | —+               |
| Можливість додавання реклами до відеопотоків | +                   | —               | —                | +                    | —                |
| Наявність маркетингових інструментів         | +                   | +               | —                | +                    | —                |
| Можливість монетизації та реклами            | +                   | +               | +                | +                    | —                |
| Підтримка клієнтів                           | +                   | +               | +—               | +                    | +—               |
| Можливість коментування відео                | —                   | —               | —                | +—                   | —                |
| Функції безпеки                              | +—                  | +               | +—               | +                    | +                |
| Потокове передавання через браузер           | —                   | +               | —                | —                    | +                |

При розробці програмного забезпечення для підтримки відеотрансляцій необхідно забезпечити такі функціональні вимоги:

- можливість авторизації та реєстрації користувачів програмного забезпечення для підтримки відеотрансляцій;
- можливість створення відеотрансляцій (для користувачів);
- можливість анонсування відеотрансляцій;
- можливість перегляду відеотрансляцій;
- можливість створення, редагування та видалення коментарів до відеотрансляцій;
- можливість збереження відеотрансляцій;
- керування користувачами, зокрема, редагування та видалення інформації про користувачів програми (для адміністратора);
- можливість блокування деяких користувачів, які викладали відеотрансляції, які не відповідають правилам програмного продукту (для адміністратора).

### **1.3 Висновки за розділом 1**

Проаналізовано проблему розробки програмного забезпечення для підтримки відеотрансляцій. Досліджено поняття про відеотрансляції. Визначено, що відеотрансляції — це розповсюдження відео- та/або аудіоконтенту серед аудиторії через Інтернет, тобто безперервне передавання аудіо- чи відеофайлів із сервера на клієнт. Досліджено основні етапи, які необхідно здійснити для організації та проведення відеотрансляцій. Відзначено, що відомі програмні засоби для підтримки відеотрансляцій, як правило, вимагають оплати для авторів відеотрансляцій, надають користувачам засоби відеоаналітики, маркетингові інструменти та дозволяють монетизувати свою діяльність. Проте у деяких наявних програмних засобах для підтримки відеотрансляцій не підтримується можливість коментування відеотрансляції, а деякі інші подібні програми є досить складними для

використання на практиці, особливо людям, які не є фахівцями з відеотрансляцій. Отже доцільно розробити програмне забезпечення для підтримки відеотрансляцій.

Визначено функціональні вимоги до розробки програмного забезпечення для підтримки відеотрансляцій, зокрема:

- можливість авторизації та реєстрації користувачів програмного забезпечення для підтримки відеотрансляцій;
- можливість створення відеотрансляцій (для користувачів);
- можливість анонсування відеотрансляцій;
- можливість перегляду відеотрансляцій;
- можливість створення, редагування та видалення коментарів до відеотрансляцій;
- можливість збереження відеотрансляцій;
- керування користувачами, зокрема, редагування та видалення інформації про користувачів програми (для адміністратора);
- можливість блокування деяких користувачів, які викладали відеотрансляції, які не відповідають правилам програмного продукту (для адміністратора).

## 2 МАТЕРІАЛИ І МЕТОДИ

### 2.1 Вибір мови програмування та інших технологій розробки

Для створення програмного забезпечення для підтримки відеотрансляцій було обрано мову програмування JavaScript [12], [13].

JavaScript є мовою сценаріїв, яка є інтерпретованою мовою програмування, яка перекладається в машинний код під час виконання коду, а не раніше. Мови сценаріїв часто використовуються для коротких сценаріїв над повними комп'ютерними програмами [12], [14].

JavaScript як мова сценаріїв використовує набір команд, які можна виконувати без необхідності компіляції. Хоча всі мови програмування є мовами програмування, не всі мови програмування є мовами сценаріїв. JavaScript як мова сценаріїв використовує програму, відому як інтерпретатор, для перекладу команд і інтерпретується безпосередньо з вихідного коду, не вимагаючи етапу компіляції [12]–[14].

JavaScript як інтерпретована мова програмування — є мовою програмування, призначеною для безпосереднього виконання вихідного коду без необхідності компілювати програму в інструкції машинної мови. Інтерпретатор виконує програму, перекладаючи оператори в одну або кілька послідовностей підпрограм перед остаточним перекладом на іншу мову, таку як машинний код. Існує два типи мов сценаріїв: на стороні сервера та на стороні клієнта. Єдина істотна відмінність між ними полягає в тому, що для обробки першого потрібен сервер. Серверні мови сценаріїв працюють на веб-сервері. Коли клієнт надсилає запит, сервер відповідає, надсилаючи вміст через HTTP. Навпаки, клієнтські мови сценаріїв працюють на стороні клієнта у веб-браузерах [12]–[14].

Перевага сценаріїв на стороні клієнта полягає в тому, що вони можуть зменшити попит на сервер, прискорюючи завантаження веб-сторінок. У той час як основна перевага сценаріїв на стороні сервера полягає в тому, що вони

не можуть переглядатися громадськістю, як сценарії на стороні клієнта [12]–[14].

Перевагами використання мови програмування JavaScript як мови сценаріїв має багато переваг перед іншими мовами програмування. По-перше, вони з відкритим кодом. Це дозволяє користувачам з усього світу брати участь у процесі вдосконалення. Серед інших переваг є такі [12]–[14]:

- немає необхідності компілювати програму;
- легкість перенесення між операційними системами;
- ефективність використання для створення вебзастосунків;
- легкість навчання та використання;
- можливість використовувати як прототип програм, заощаджуючи час на тестових проектах.

Одним із недоліків використання мови програмування JavaScript як мови сценаріїв є те, що деякі компанії не хочуть, щоб їхні сценарії мали можливість бачити та аналізувати всі користувачі, тому вони використовують сценарії на стороні сервера, щоб уникнути їх публікації для громадськості. Крім того, встановлення програми інтерпретатора може бути складним. Нарешті, іноді програмне забезпечення у вигляді сценаріїв працює повільніше, ніж відкомпільовані програми [12]–[14].

JavaScript використовується програмістами для створення динамічного та інтерактивного веб-вмісту, наприклад програм і браузерів. JavaScript настільки популярний, що це найпоширеніша мова програмування у світі, яка використовується як мова програмування на стороні клієнта на переважній більшості усіх вебсайтів. Клієнтські мови – це ті, дія яких відбувається на комп'ютері користувача, а не на сервері [12]–[14].

JavaScript досить універсальний, щоб його можна було використовувати для різноманітних програм, таких як програмне забезпечення, апаратне забезпечення та сервери. JavaScript найбільш відомий як веб-мова, оскільки він специфічний для веб-браузера. Веб-браузер може

природно розуміти мову так само, як носій мови може природно розуміти англійську [12]–[14].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки відеотрансляцій наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки відеотрансляцій

| Критерій порівняння мов програмування                                                       | Мова програмування |     |     |
|---------------------------------------------------------------------------------------------|--------------------|-----|-----|
|                                                                                             | JavaScript         | PHP | C++ |
| Зручність використання для створення програмного забезпечення для підтримки відеотрансляцій | +                  | +—  | —   |
| Легкість вивчення                                                                           | +                  | +   | +—  |
| Необхідність у компіляції розробленої програми                                              | —                  | —   | +   |
| Бібліотеки та засоби захисту інформації                                                     | +                  | +   | +—  |
| Ефективність для розробки клієнт-серверних застосунків                                      | +                  | +   | +—  |

Таким чином, для розробки програмного забезпечення для підтримки відеотрансляцій обрано мову програмування JavaScript.

## 2.2 Вибір фреймворку для створення програмного забезпечення

В якості фреймворку для створення програмного забезпечення для підтримки відеотрансляцій було використано Node.js [15]–[16].

Node.js — це середовище виконання JavaScript. Середовище виконання Node.js містить усе необхідне для запуску програми, написаної на JavaScript.

Node.js з'явився, коли початкові розробники JavaScript розширили його з того, що можна просто запускати в браузері, до того, що можна запускати як окрему програму на вашому комп'ютері.

JavaScript за допомогою Node.js може робити те, що інші мови сценаріїв, такі як Python.

JavaScript браузера, і Node.js працюють на механізмі виконання V8 JavaScript. Цей механізм використовує код JavaScript і перетворює його на швидший машинний код. Машинний код — це код низького рівня, який комп'ютер може запускати без необхідності його попередньої інтерпретації.

Основні особливості Node.js:

- Node.js — це середовище виконання JavaScript, створене на движку JavaScript V8 Chrome;
- Node.js використовує неблокуючу модель введення-виведення, керовану подіями, що робить його легким і ефективним.
- екосистема пакетів Node.js, є найбільшою бібліотечною екосистемою з відкритим кодом у світі.

Node.js має велику кількість бібліотек, створених спільнотою програмістів, які вирішують більшість типових проблем, пов'язаних з розробкою вебзастосунків. Модуль вузла у Node.js — це багаторазовий блок коду, існування якого випадково не впливає на інший код.

Node.js — це серверна платформа, побудована на движку Google Chrome JavaScript (V8 Engine). Node.js — це кросплатформне середовище виконання з відкритим кодом для розробки серверних і мережевих програм. Програми Node.js написані на JavaScript і можуть запускатися в середовищі виконання Node.js в OS X, Microsoft Windows і Linux.

Node.js також надає багату бібліотеку різноманітних модулів JavaScript, що значно спрощує розробку веб-додатків за допомогою Node.js. Програміст має можливість писати власні модулі та використовувати їх у різних програмах. Node.js можна використовувати без будь-якої іншої інсталяції

Особливостями фреймворку Node.js є такі:

– асинхронність і керованість подіями – усі API бібліотеки Node.js є асинхронними, тобто неблокуючими. По суті, це означає, що сервер на основі Node.js ніколи не очікує від API повернення даних. Сервер переходить до наступного API після його виклику, а механізм сповіщень Events of Node.js допомагає серверу отримати відповідь від попереднього виклику API;

– дуже велика швидкість роботи. Бібліотека Node.js, створена на платформі Google Chrome V8 JavaScript Engine, дуже швидко виконує код;

– однопоточність та велика масштабованість – Node.js використовує однопотокову модель із циклом подій. Механізм подій допомагає серверу відповідати неблокуючим способом і робить сервер високомасштабованим на відміну від традиційних серверів, які створюють обмежені потоки для обробки запитів. Node.js використовує одну потокову програму, і та сама програма може обслуговувати набагато більшу кількість запитів, ніж традиційні сервери, такі як HTTP-сервер Apache;

– відсутність використання буферизації – програми Node.js ніколи не буферизуватимуть дані. Ці програми просто експортують дані порціями;

– ліцензія – Node.js випущено згідно з ліцензіями MIT.

NodeJS в основному розроблено за допомогою JavaScript, CoffeeScript і C++. NodeJS — це асинхронне середовище виконання JavaScript, кероване подіями, яке може запускати код JavaScript поза веб-браузером. NodeJS побудовано на JavaScript, тому він надзвичайно легкий і неймовірно ефективний.

NodeJS є досить ефективним при розробці програм, які потребують роботи зі складною та інтенсивною обробкою даних. NodeJS може легко обробляти складні дані, тому багато програм, які потребують інтенсивної обробки даних, використовують NodeJS.

Обґрунтування вибору фреймворку для створення програмного забезпечення для підтримки відеотрансляцій наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору фреймворку для створення програмного забезпечення для підтримки відеотрансляцій

| Критерій порівняння фреймворків                            | Фреймворк |        |             |
|------------------------------------------------------------|-----------|--------|-------------|
|                                                            | Node.js   | django | Spring Boot |
| Швидкість роботи                                           | +         | +—     | +—          |
| Асинхронність                                              | +         | —      | —           |
| Можливість використання для розробки програм на JavaScript | +         | —      | +           |
| Універсальність                                            | +         | +—     | +           |
| Масштабованість                                            | +         | +      | +—          |

Таким чином, для створення програмного забезпечення для підтримки відеотрансляцій обрано фреймворк Node.js.

### 2.3 Розробка моделі взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій

З метою можливості розробки програмного забезпечення для підтримки відеотрансляцій створимо відповідну модель у нотації UML у вигляді діаграми прецедентів. Розроблена модель дозволить описати поведінку програмного забезпечення підтримки відеотрансляцій в термінах послідовності дій.

При розробленні відповідної моделі враховуємо те, що користувачі програмного забезпечення підтримки відеотрансляцій (звичайний користувач та адміністратор) маю декілька можливостей.

Основний успішний сценарій користувача при роботі з відеотрансляціями можна описати таким чином:

- користувач авторизується;

- після авторизації користувача сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- користувач бачить відповідну сторінку та обирає потрібну дію (перегляд трансляції, створення нової відеотрансляції, видалення власних трансляцій або оновлення інформації);
- користувач виконує відповідну дію;
- користувач завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Опишемо основні дії користувача стосовно основного функціоналу програмного забезпечення підтримки відеотрансляцій.

Основний успішний сценарій користувача при перегляді відеотрансляцій можна описати таким чином:

- користувач авторизується;
- після авторизації користувача сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- користувач бачить відповідну сторінку та обирає відеотрансляцію для перегляду;
- користувач переглядає обрану відеотрансляцію;
- користувач завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Основний успішний сценарій користувача при створенні відеотрансляцій можна описати таким чином:

- користувач авторизується;
- після авторизації користувача сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- користувач бачить відповідну сторінку та обирає можливість створення відеотрансляції;
- користувач створює нову відеотрансляцію та здійснює стрімінове відеомовлення;

- користувач завершує відеотрансляцію та завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Основний успішний сценарій користувача при видаленні власних відеотрансляцій можна описати таким чином:

- користувач авторизується;
- після авторизації користувача сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- користувач бачить відповідну сторінку та обирає можливість видалення власної відеотрансляції;
- користувач видаляє обрану відеотрансляцію;
- користувач завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Основний успішний сценарій користувача при оновленні власних відеотрансляцій можна описати таким чином:

- користувач авторизується;
- після авторизації користувача сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- користувач бачить відповідну сторінку та обирає можливість оновлення власної відеотрансляції;
- користувач оновлює обрану відеотрансляцію (додає або змінює інформацію про відеотрансляцію, або завантажує нову трансляцію замість наявної);
- користувач завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Основний успішний сценарій адміністратора при роботі з відеотрансляціями можна описати таким чином:

- адміністратор авторизується;
- після авторизації адміністратора сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;

- адміністратор бачить відповідну сторінку та обирає потрібну дію (оновлення наявної трансляції, видалення відеотрансляції, блокування користувача);

- адміністратор виконує відповідну дію;
- адміністратор завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Основний успішний сценарій адміністратора при оновленні відеотрансляцій можна описати таким чином:

- адміністратор авторизується;
- після авторизації адміністратора сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- адміністратор бачить відповідну сторінку та обирає можливість оновлення відеотрансляцій користувачів;
- адміністратор оновлює обрану відеотрансляцію (додає або змінює інформацію про відеотрансляцію);
- адміністратор завершує роботу з програмним забезпеченням підтримки відеотрансляцій.

Дії з видалення відеотрансляцій адміністратора аналогічні тим, які виконує звичайний користувач.

Основний успішний сценарій адміністратора при видаленні користувачів можна описати таким чином:

- адміністратор авторизується;
- після авторизації адміністратора сервер програмного забезпечення підтримки відеотрансляцій у відповідь на запит повертає основну сторінку;
- адміністратор бачить відповідну сторінку та обирає можливість блокування користувачів;
- адміністратор обирає користувача та блокує його;
- адміністратор завершує роботу з програмним забезпеченням підтримки відеотрансляцій.



## 2.4 Висновки за розділом 2

Для створення програмного забезпечення для підтримки відеотрансляцій обрано мову програмування JavaScript, яка є ефективною для розробки вебзастосунків.

Як фреймворк для створення програмного забезпечення для підтримки відеотрансляцій обрано Node.js, що є досить ефективним при розробці програм, які потребують роботи зі складною та інтенсивною обробкою даних.

Створено модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у нотації UML, та дозволяє розроблювати програмне забезпечення для підтримки процесу відеострімінгу.

### 3 ОПИС ПРОГРАМИ

#### 3.1 Структура програмного забезпечення для підтримки відеотрансляцій

Структуру програмного забезпечення для підтримки відеотрансляцій подамо у вигляді діаграм в нотації UML (рис. 3.1 – 3.3).

Загальна діаграма класів програмного забезпечення для підтримки відеотрансляцій зображена на рис. 3.1.

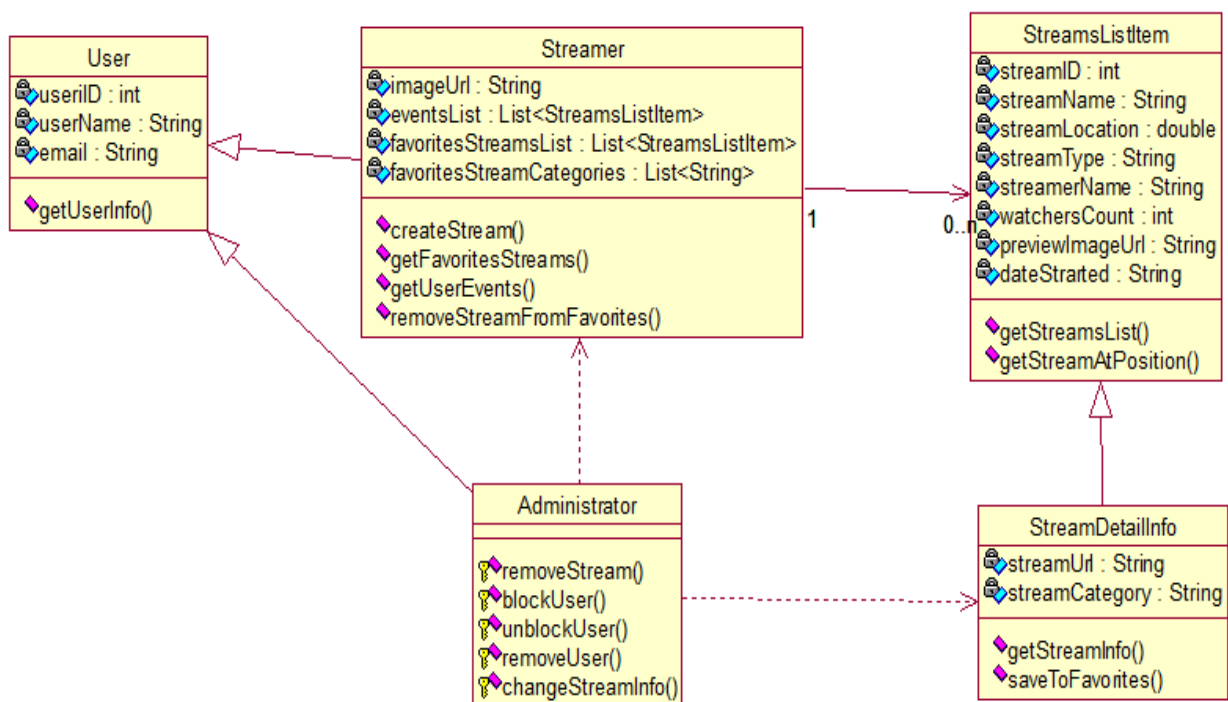


Рисунок 3.1 – Діаграма класів програмного забезпечення для підтримки відеотрансляцій у нотації UML

Як видно з діаграми, основними класами системи є: Стрімер (Streamer), Користувач (User), Відеотрансляції (StreamsListItem), Адміністратор (Administrator), Інформація про відеотрансляцію (StreamDetailInfo).

Діаграма пакетів програмного забезпечення для підтримки відеотрансляцій зображена на рис. 3.2.

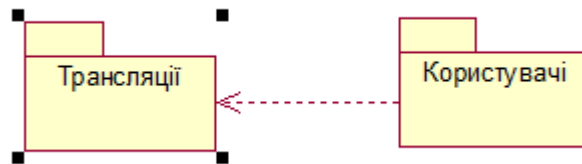


Рисунок 3.2 – Діаграма пакетів програмного забезпечення для підтримки відеотрансляцій у нотації UML

Діаграма компонентів програмного забезпечення для підтримки відеотрансляцій зображена на рис. 3.3.

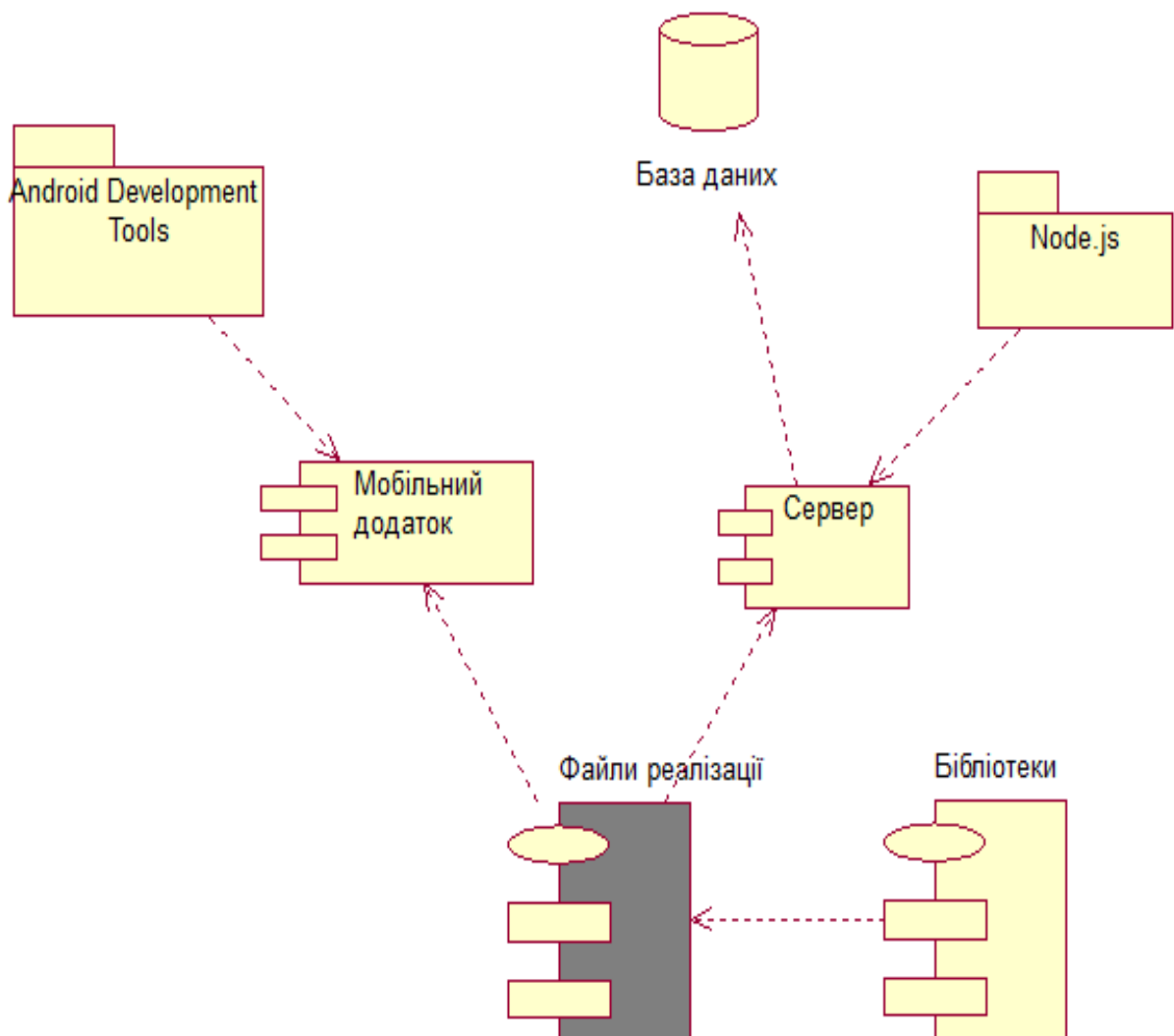


Рисунок 3.3 – Діаграма компонентів програмного забезпечення для підтримки відеотрансляцій у нотації UML

Як видно з рисунку, основними компонентами є мобільний застосунок, сервер, фреймворк Node.js, база даних, бібліотеки та ін.

### 3.2 Функціонування програмного забезпечення для підтримки відеотрансляцій

Загальну схему функціонування програмного забезпечення для підтримки відеотрансляцій подамо у вигляді діаграми станів, зображеної на рис. 3.4.

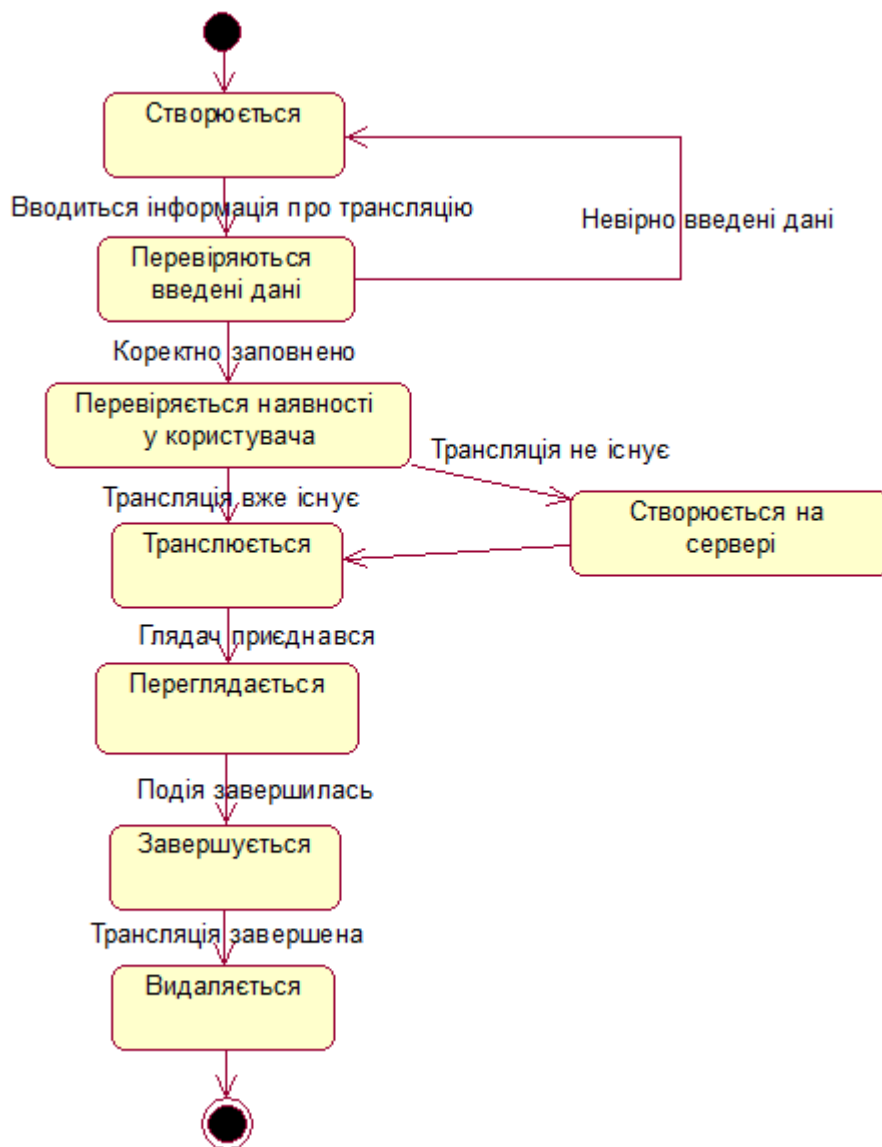


Рисунок 3.4 – Загальна схема функціонування програмного забезпечення для підтримки відеотрансляцій у вигляді діаграми станів, поданої у нотації UML

Діаграма діяльності програмного забезпечення для підтримки відеотрансляцій зображена на рис. 3.5.

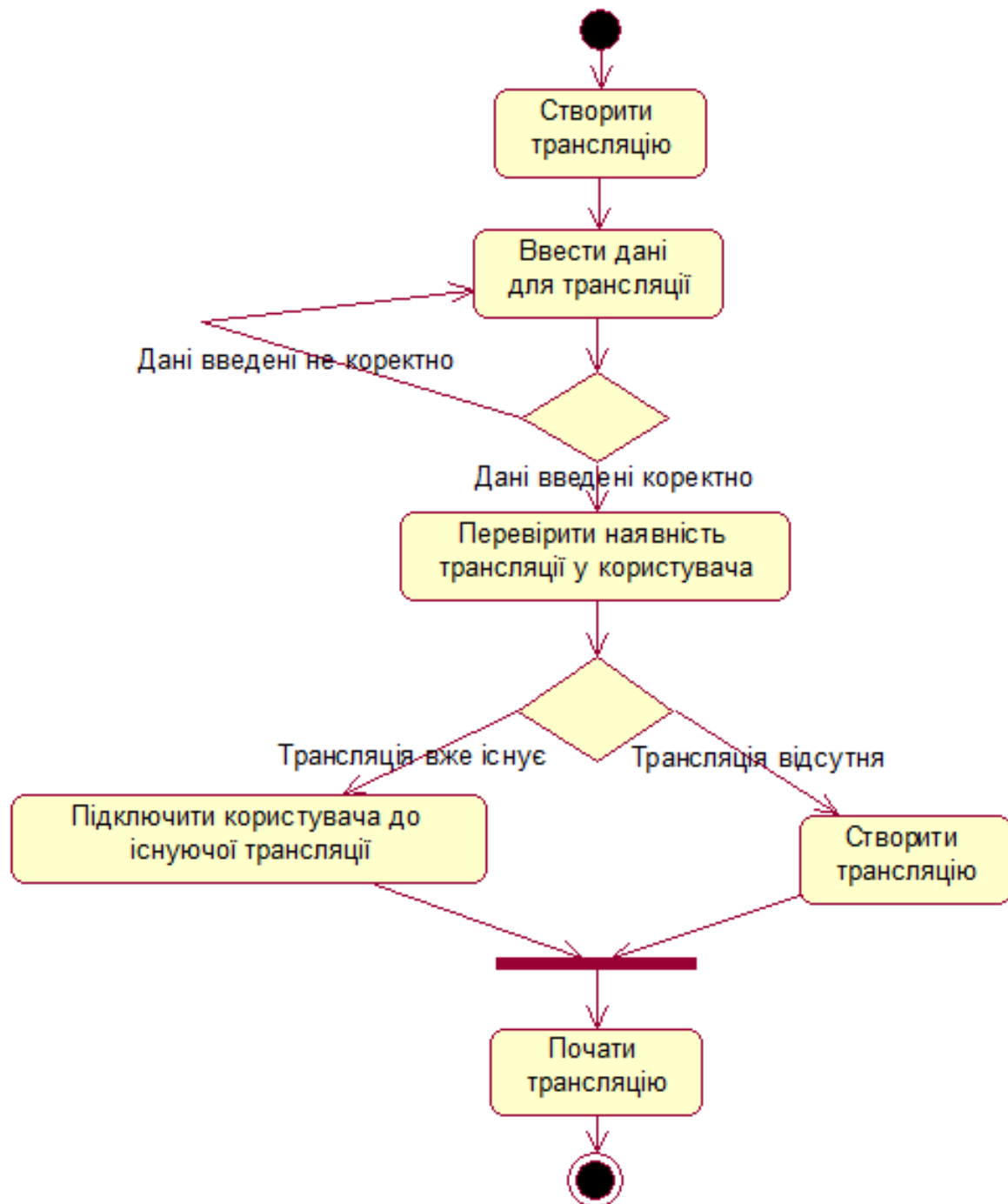


Рисунок 3.5 – Діаграма діяльності програмного забезпечення для підтримки відеотрансляцій у нотації UML

Діаграму кооперацій у процесі стрімінгу за допомогою програмного забезпечення для підтримки відеотрансляцій у нотації UML зображено на рис. 3.6.

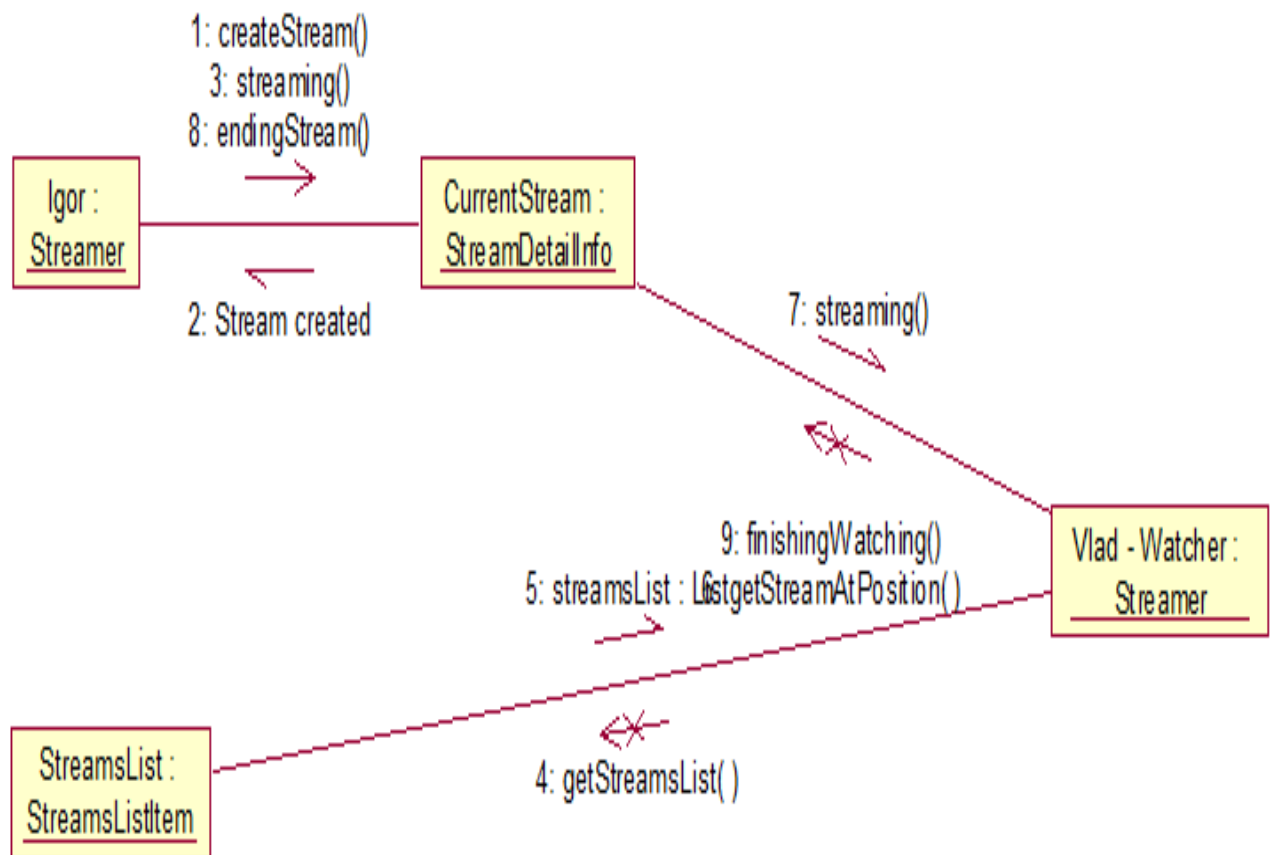


Рисунок 3.6 – Діаграма кооперацій програмного забезпечення для підтримки відеотрансляцій у нотації UML

Діаграма послідовності програмного забезпечення для підтримки відеотрансляцій у нотації UML зображена на рис. 3.7.

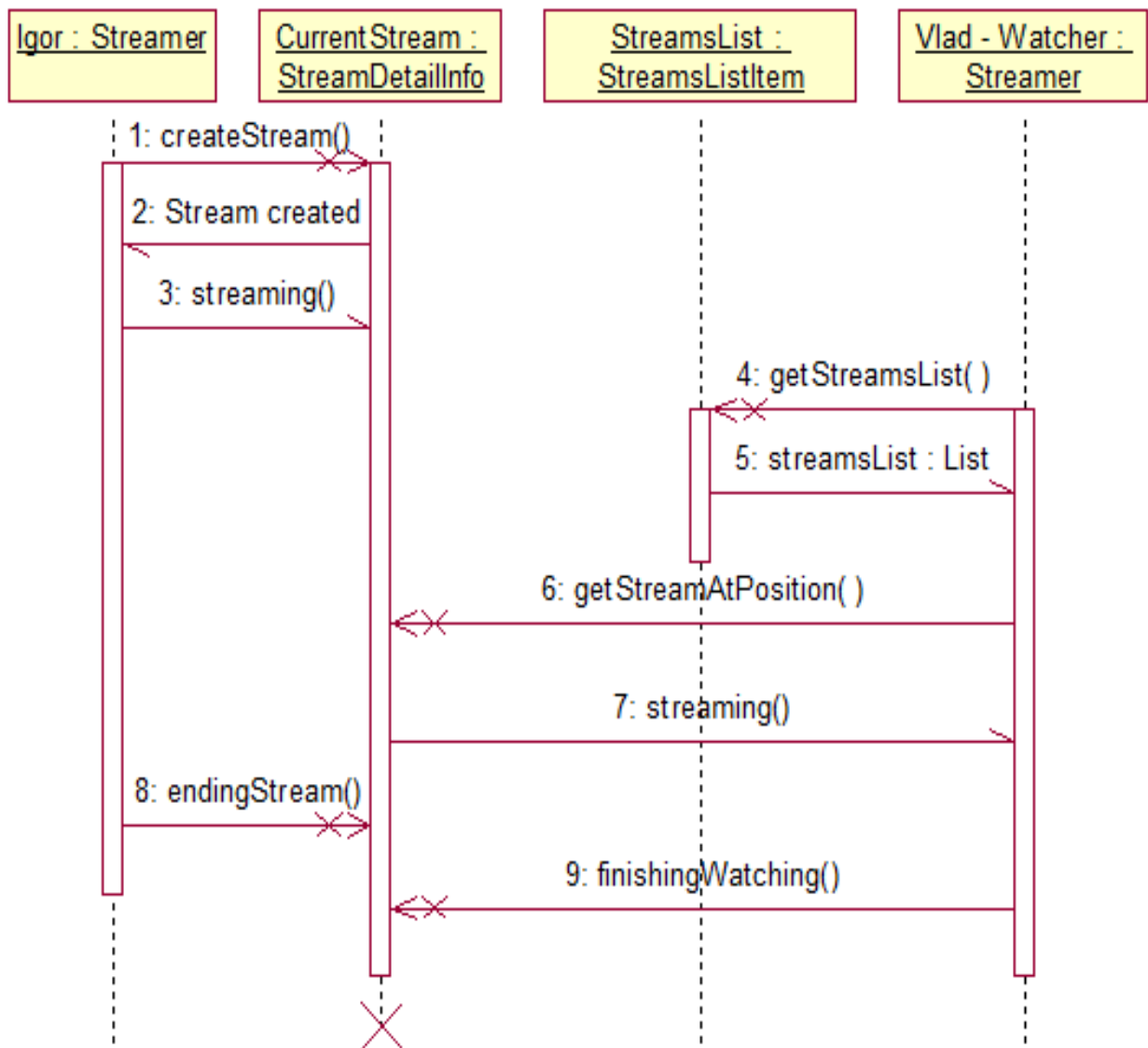


Рисунок 3.7 – Діаграма послідовностей програмного забезпечення для підтримки відеотрансляцій у нотації UML

### 3.3 Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки відеотрансляцій

При розробленні програмного забезпечення для підтримки відеотрансляцій враховувалися вимоги технічного завдання, а також рішення прийняті у процесі проєктування. Враховуючи вищезазначене, було створено декілька екранних форм, що дозволяють забезпечити виконання вимог,

поставлених у технічному завданні на виконання розробки програмного забезпечення для підтримки відеотрансляцій.

Форму для авторизації користувача програмного забезпечення для підтримки відеотрансляцій наведено на рис. 3.8.

The image displays two mobile application screens for 'STREAMCAST'. Both screens have a white background with a light gray pattern of small geometric shapes. At the top of each screen is a black status bar showing signal strength, battery level, and the time '12:30'. Below the status bar is a header section with a blue play button icon inside a square frame, followed by the word 'STREAMCAST' in blue. Underneath the header are two links: 'SIGN IN' and 'SIGN UP', with 'SIGN IN' being underlined. The left screen is the 'Sign In' page. It features a light gray input field for 'Email Address' containing the text 'Lorem isum@gmail.com', followed by a 'Password' input field. Below these is a solid blue button with the text 'SIGN IN' in white. At the bottom of the form is a link 'Forgot Password?'. The right screen is the 'Sign Up' page. It features a light gray input field for 'Email Address', followed by 'Password' and 'Confirm Password' input fields. Below these is a solid blue button with the text 'SIGN UP' in white. At the bottom of the form is a link 'Already have an account? Sign In'. Both screens have a black Android navigation bar at the bottom with white icons for back, home, and recent apps.

Рисунок 3.8 – Форми авторизації користувача програмного забезпечення для підтримки відеотрансляцій

Головне вікно та вікно створення трансляцій програмного забезпечення для підтримки відеотрансляцій відображені на рис. 3.9.

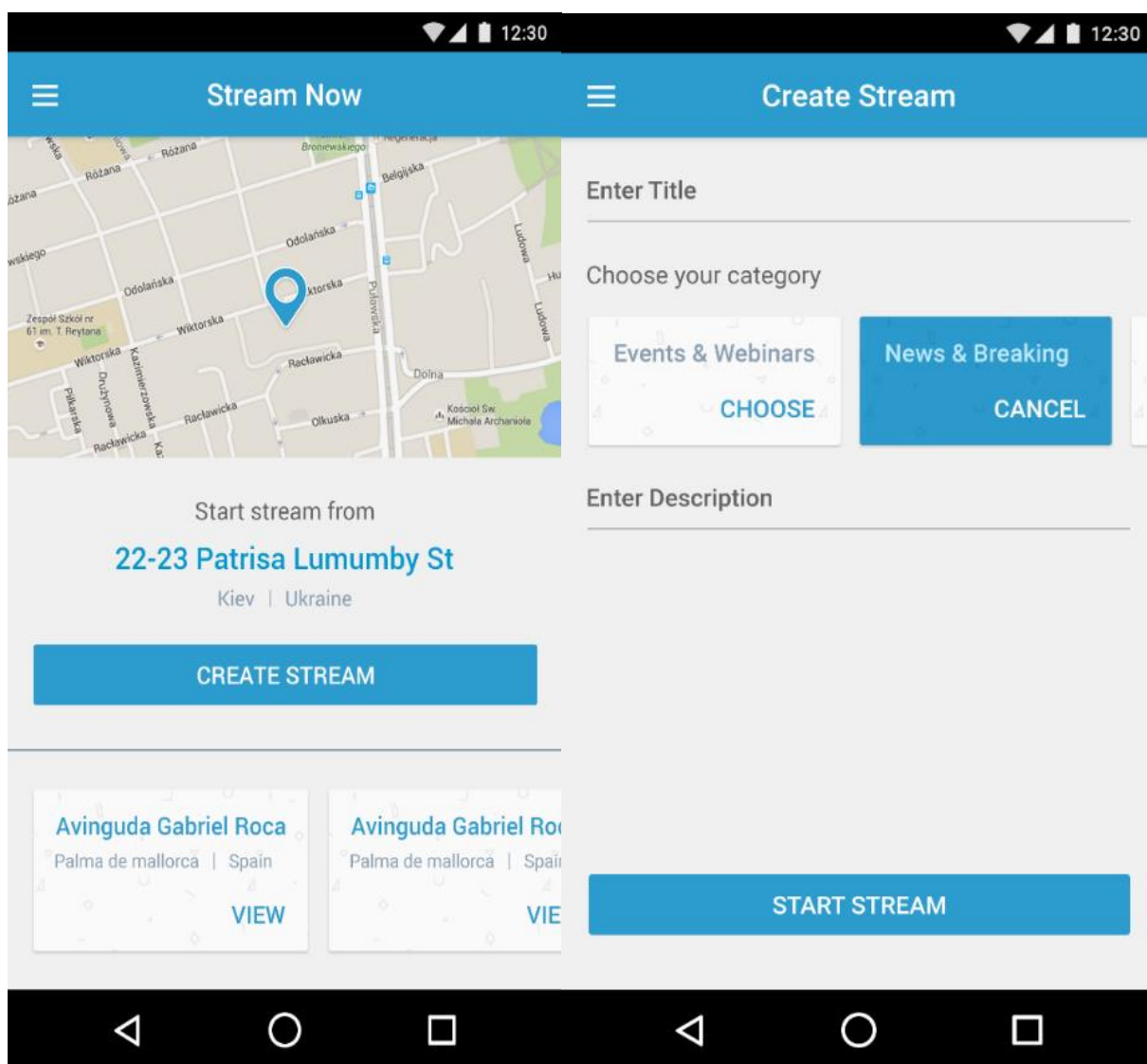


Рисунок 3.9 – Головне вікно та вікно створення трансляцій програмного забезпечення для підтримки відеотрансляцій

Вікно відображення створених трансляцій за допомогою програмного забезпечення для підтримки відеотрансляцій відображено на рис. 3.10.

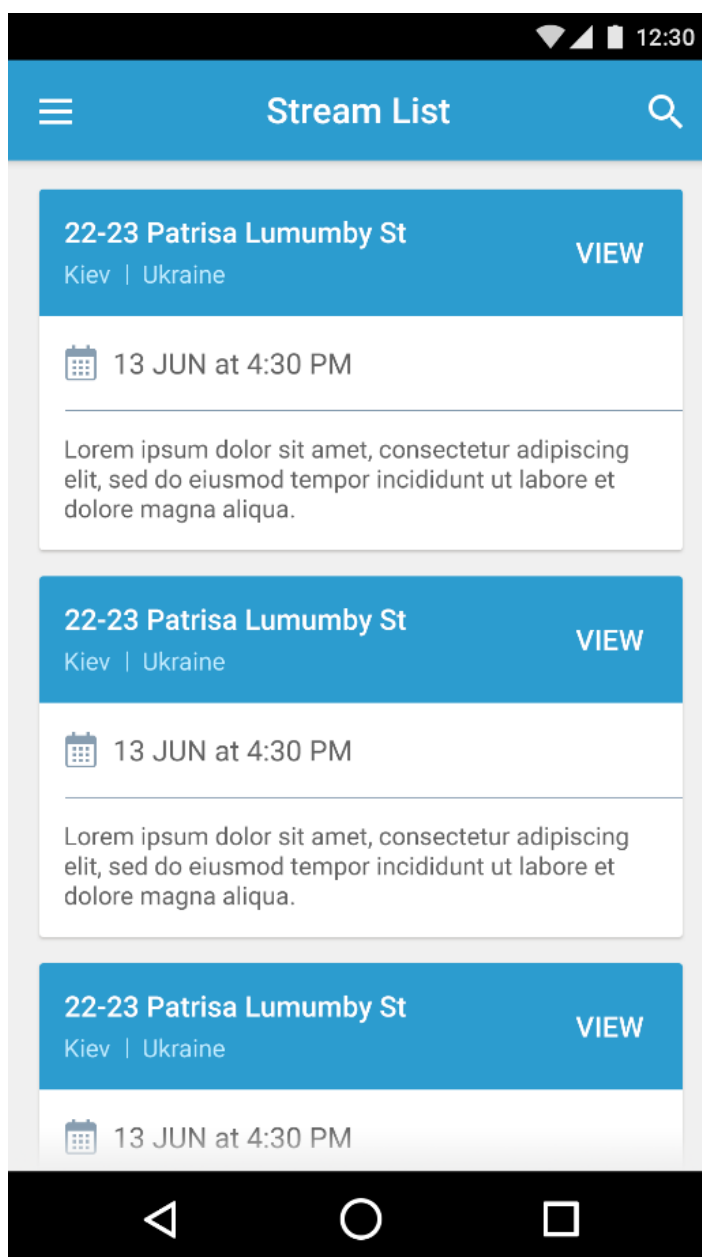


Рисунок 3.10 – Вікно відображення створених трансляцій за допомогою програмного забезпечення для підтримки відеотрансляцій

### 3.4 Опис розроблених модулів програмного забезпечення для підтримки відеотрансляцій

З метою подання опису розроблених модулів програмного забезпечення для підтримки відеотрансляцій надамо інформацію про фізичну структуру розробленої програми.

Програмне забезпечення для підтримки відеотрансляцій складається з двох частин:

- серверна частина програмного забезпечення для підтримки відеотрансляцій;
- мобільна частина програмного забезпечення для підтримки відеотрансляцій.

Фізична структура серверної частини програмного забезпечення для підтримки відеотрансляцій відображена на рис. 3.11.

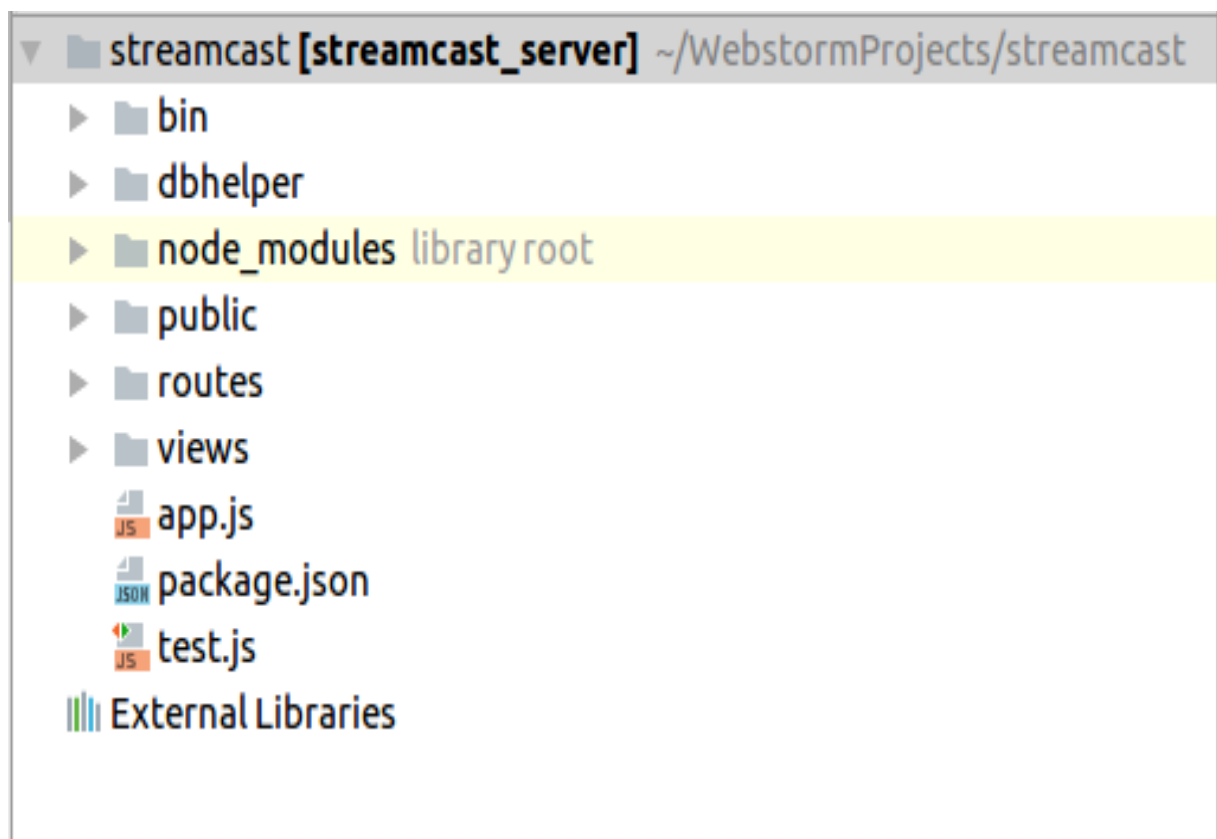


Рисунок 3.11 – Фізична структура серверної частини програмного забезпечення для підтримки відеотрансляцій

Опис основних програмних модулів серверної частини програмного забезпечення для підтримки відеотрансляцій наведено у табл. 3.1.

Таблиця 3.1 – Опис основних програмних модулів серверної частини програмного забезпечення для підтримки відеотрансляцій

| Модуль       | Призначення                                                                                                                                             |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| bin          | модуль створення та налаштування серверу програмного забезпечення для підтримки відеотрансляцій                                                         |
| dbhelper     | модуль для роботи з базою даних програмного забезпечення для підтримки відеотрансляцій                                                                  |
| node_modules | модуль для роботи з бібліотеками                                                                                                                        |
| public       | модуль зображень для розширень, містить інформацію про каскадні таблиці стилів, сторонні скрипти програмного забезпечення для підтримки відеотрансляцій |
| routes       | модуль з кодом серверної частини програмного забезпечення для підтримки відеотрансляцій                                                                 |
| views        | модуль графічного представлення сторінок програмного забезпечення для підтримки відеотрансляцій                                                         |

Фізична структура мобільної частини програмного забезпечення для підтримки відеотрансляцій відображена на рис. 3.12.

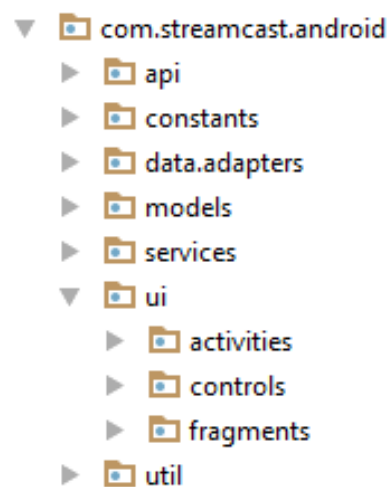


Рисунок 3.12 – Фізична структура мобільної частини програмного забезпечення для підтримки відеотрансляцій

Опис основних програмних модулів мобільної частини програмного забезпечення для підтримки відеотрансляцій наведено у табл. 3.2.

Таблиця 3.2 – Опис основних програмних модулів мобільної частини програмного забезпечення для підтримки відеотрансляцій

| Папка         | Призначення                                                                                                          |
|---------------|----------------------------------------------------------------------------------------------------------------------|
| api           | модуль для роботи з серверною частиною програмного забезпечення для підтримки відеотрансляцій                        |
| constants     | модуль констант програмного забезпечення для підтримки відеотрансляцій                                               |
| data.adapters | модуль для роботи з даними в елементах інтерфейсу користувача програмного забезпечення для підтримки відеотрансляцій |
| models        | модуль для роботи з користувацькими типами даних програмного забезпечення для підтримки відеотрансляцій              |
| services      | модуль для роботи зі сторонніми сервісами програмного забезпечення для підтримки відеотрансляцій                     |
| ui.activities | модуль активностей інтерфейсу користувача програмного забезпечення для підтримки відеотрансляцій                     |
| ui.controls   | модуль функціоналу деяких елементів інтерфейсу користувача програмного забезпечення для підтримки відеотрансляцій    |
| ui.fragments  | модуль фрагментів інтерфейсу користувача програмного забезпечення для підтримки відеотрансляцій                      |
| util          | модуль функціоналу стандартних утиліт програмного забезпечення для підтримки відеотрансляцій                         |

Таким чином, визначено, що програмне забезпечення для підтримки відеотрансляцій складається з серверної та мобільної частин. Подано опис

розроблених модулів серверної та мобільної частин програмного забезпечення для підтримки відеотрансляцій.

### **3.5 Висновки за розділом 3**

Розроблено програмне забезпечення для підтримки відеотрансляцій. Запропоновано структуру програмного забезпечення для підтримки відеотрансляцій, яку подано за допомогою структурних діаграм в нотації UML.

Визначено, що основними класами системи є: Стрімер (Streamer), Користувач (User), Відеотрансляції (StreamsListItem), Адміністратор (Administrator), Інформація про відеотрансляцію (StreamDetailInfo). Основними компонентами є мобільний застосунок, сервер, фреймворк Node.js, база даних, бібліотеки та ін.

Описано функціонування програмного забезпечення для підтримки відеотрансляцій за допомогою моделей у нотації UML основі діаграм поведінки, діаграм кооперації та діаграм послідовності.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки відеотрансляцій.

Визначено, що програмне забезпечення для підтримки відеотрансляцій складається з серверної та мобільної частин. Подано опис розроблених модулів серверної та мобільної частин програмного забезпечення для підтримки відеотрансляцій.

## **4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ**

### **4.1 Призначення й умови застосування програми**

Програмне забезпечення для підтримки відеотрансляцій призначено для надання користувачу можливості здійснення та перегляду стрімінгових відео. Програмне забезпечення для підтримки відеотрансляцій написано на мові JavaScript. Як фреймворк для створення програмного забезпечення для підтримки відеотрансляцій обрано Node.js,

Програмне забезпечення для підтримки відеотрансляцій забезпечує виконання таких функцій:

- можливість авторизації та реєстрації користувачів програмного забезпечення для підтримки відеотрансляцій;
- можливість створення відеотрансляцій (для користувачів);
- можливість анонсування відеотрансляцій;
- можливість перегляду відеотрансляцій;
- можливість створення, редагування та видалення коментарів до відеотрансляцій;
- можливість збереження відеотрансляцій;
- керування користувачами, зокрема, редагування та видалення інформації про користувачів програми (для адміністратора);
- можливість блокування деяких користувачів, які викладали відеотрансляції, які не відповідають правилам програмного продукту (для адміністратора).

Для експлуатації програмного забезпечення для підтримки відеотрансляцій необхідно таке апаратне забезпечення:

- сервер із процесором з тактовою частотою 3ГГц і вище;
- об'єм оперативної пам'яті на сервері – не менше 512 Мб;
- вільне місце на жорсткому диску сервера – 2 Гб;
- монітор та маніпулятор типу «миша»;

– клієнтський пристрій (мобільний телефон, ноутбук, комп'ютер) з можливістю виходу в інтернет.

Функціональні характеристики розробленого програмного забезпечення для підтримки відеотрансляцій наведено у технічному завданні (додаток А). Опис програмного забезпечення для підтримки відеотрансляцій та текст програмного забезпечення для підтримки відеотрансляцій наведені у додатках Б та В.

## **4.2 Характеристики програми**

Програмне забезпечення для підтримки відеотрансляцій може працювати цілодобово, в залежності від можливостей електроживлення сервера.

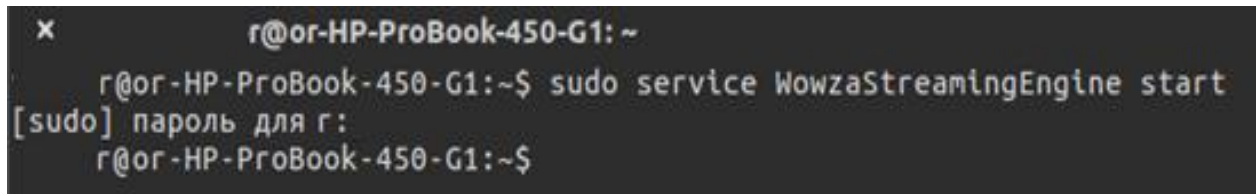
Програмне забезпечення для підтримки відеотрансляцій не потребує особливих умов експлуатації, окрім наявності серверу, мобільного (або іншого) пристрою користувача та можливості електроживлення. Інтерфейс програмного забезпечення для підтримки відеотрансляцій не потребує особливої кваліфікації користувача.

## **4.3 Розгортання вебсерверу для функціонування програмного забезпечення для підтримки відеотрансляцій**

Для розробки вебсерверу програмного забезпечення для підтримки відеотрансляцій було використано фреймворк Node.js. Для розгортання вебсерверу програмного забезпечення для підтримки відеотрансляцій його необхідно запустити за допомогою середовища розробки, зокрема, для цього можна використовувати WebStorm [17].

Для встановлення локальної версії двигуна Wowza Streaming Engine, необхідно перейти за посиланням [wowza.com](http://wowza.com), завантажити відповідну версію локального вебсерверу та здійснити його установку на комп'ютер. Після цього

встановлення програмного забезпечення на комп'ютер, необхідно запустити локальний вебсервер (рис. 4.1).



```
x      r@or-HP-ProBook-450-G1: ~  
r@or-HP-ProBook-450-G1:~$ sudo service WowzaStreamingEngine start  
[sudo] пароль для r:  
r@or-HP-ProBook-450-G1:~$
```

Рисунок 4.1 – Запуск локального вебсервера з командного рядка

Після запуску локального вебсервера необхідно створити користувача, використовуючи для цього панель адміністратора вебсервера Wowza Streaming Engine (рис. 4.2).

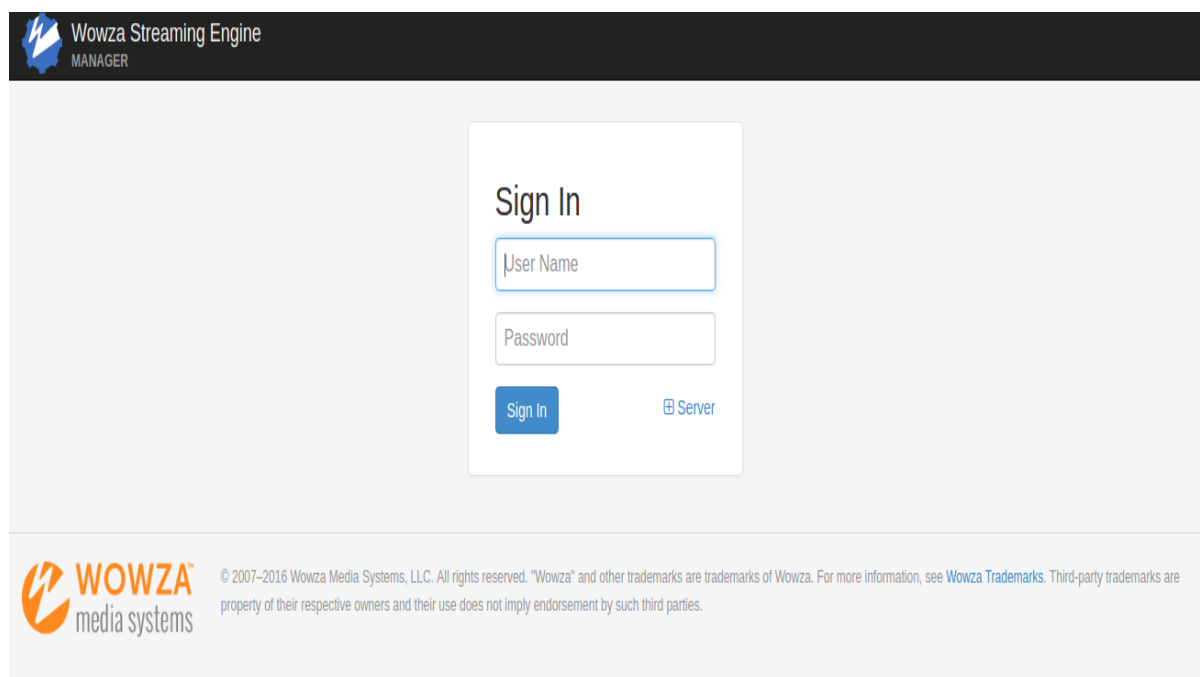


Рисунок 4.2 – Створення користувача у панелі адміністратора вебсервера

Далі необхідно виконати конфігурування вебсерверу (рис. 4.3) та здійснити його налаштування для можливості забезпечення відеотрансляцій (рис.4.4).

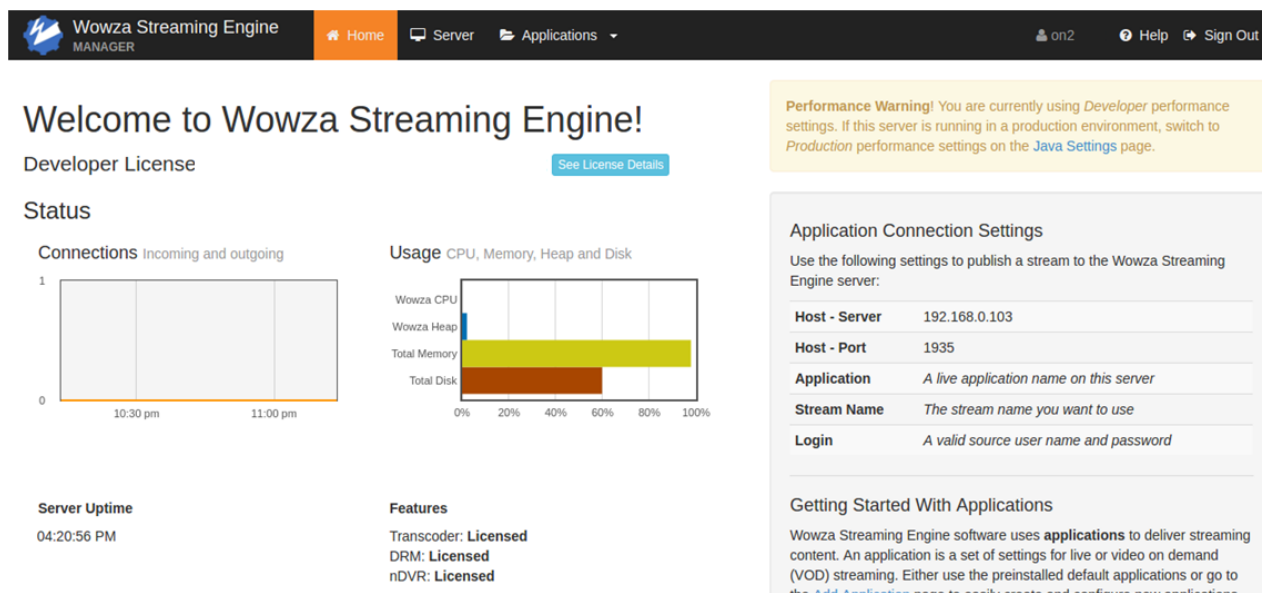


Рисунок 4.3 – Конфігурація вебсерверу

**live**  
Live Single Server or Origin

\* = Required field [Hide Help](#)

[Save](#) [Cancel](#)

**Application Description**

Default application for live streaming created when Wowza Streaming Engine is installed. Use this application with its default configuration or modify the configuration as needed. You can also copy it to create another live application.

**Playback Types \***

- ☒ MPEG-DASH
- ☒ Apple HLS
- ☒ Adobe RTMP
- ☒ Adobe HDS
- ☒ Microsoft Smooth Streaming
- ☒ RTSP/RTP

**Options**

- ☐ Low-latency stream (ideal for chat applications)
- ☐ Record all incoming streams
- ☒ Cross-origin resource sharing (CORS) (for HTTP-based streams)

**Streaming File Directory \***

- ☒ Use default

**Application Description**

Enter a description that will help you to remember details about your application.

**Playback Types**

Enable streaming to specific endpoints by selecting one or more of the following **Playback Types**:

- MPEG-DASH.** Stream to DASH clients via Dynamic Adaptive Streaming over HTTP (DASH).
- Apple HLS.** Stream to Apple iOS devices and Android devices via Apple HTTP Live Streaming protocol.
- Adobe RTMP.** Stream to Adobe Flash Player via Real Time Messaging Protocol.
- Adobe HDS.** Stream to Adobe Flash Player via Adobe HTTP Dynamic Streaming protocol.
- Microsoft Smooth Streaming.** Stream to Smooth Streaming clients via Microsoft Smooth Streaming protocol.
- RTSP/RTP.** Stream to RTSP/RTP-

**SELECTED APPLICATION**

live ▼

- Monitoring
- Sources (Live)
- Stream Files
- Incoming Streams
- Wowza Player
- Stream Targets
- Source Security
- Playback Security
- SMIL Files
- nDVR
- Transcoder
- DRM

**LIVE APPLICATIONS**

- live

**VOD APPLICATIONS**

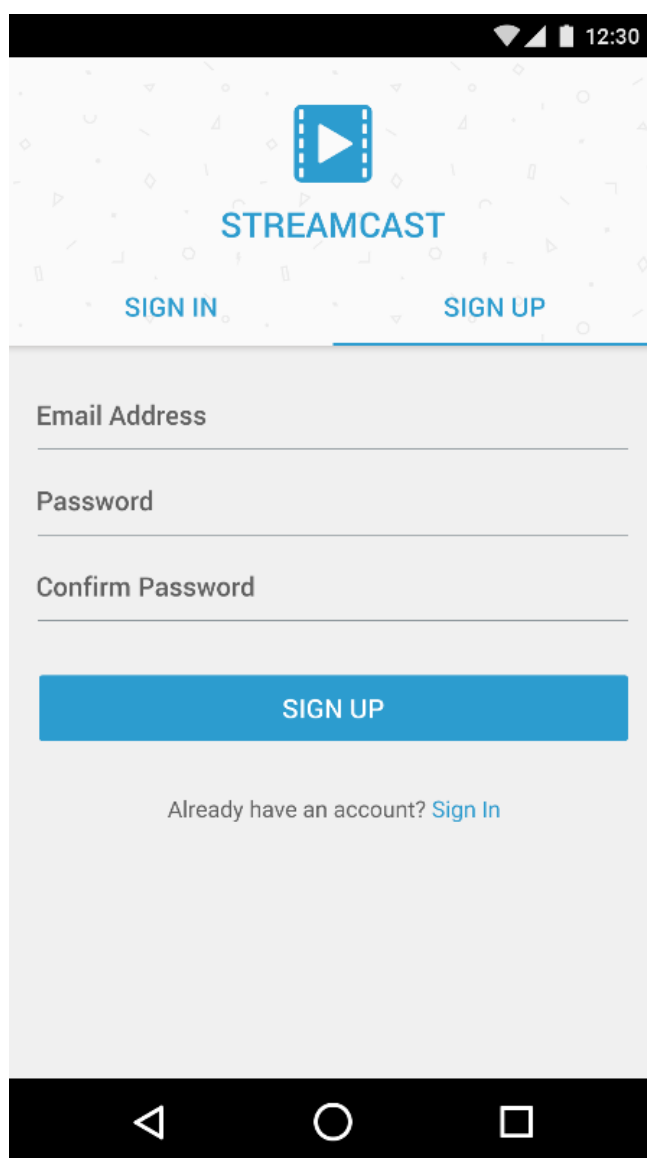
- vod

Рисунок 4.4 – Налаштування вебсерверу для можливості забезпечення відеотрансляцій

#### 4.4 Опис роботи з програмою та результати тестування програмного забезпечення для підтримки відеотрансляцій

Для роботи з програмою користувач повинен здійснити запуск її клієнтської (мобільної) версії за допомогою свого пристрою.

Після запуску програмного забезпечення для підтримки відеотрансляцій перший раз користувачу необхідно зареєструватися (рис. 4.5).



The screenshot displays the registration interface of the StreamCast application on a mobile device. At the top, the status bar shows the time as 12:30. The app's header features a blue play button icon and the word 'STREAMCAST' in blue capital letters. Below the header, there are two links: 'SIGN IN' and 'SIGN UP'. The main form area contains three input fields labeled 'Email Address', 'Password', and 'Confirm Password'. A prominent blue button labeled 'SIGN UP' is positioned below the input fields. At the bottom of the form, there is a link that reads 'Already have an account? Sign In'. The entire interface is set against a light gray background with a subtle pattern of small geometric shapes.

Рисунок 4.5 – Форма реєстрації користувача програмного забезпечення для підтримки відеотрансляцій

У випадку, якщо користувач вже зареєстрований у програмному забезпеченні для підтримки відеотрансляцій при вході в систему необхідно пройти авторизацію (рис. 4.6).

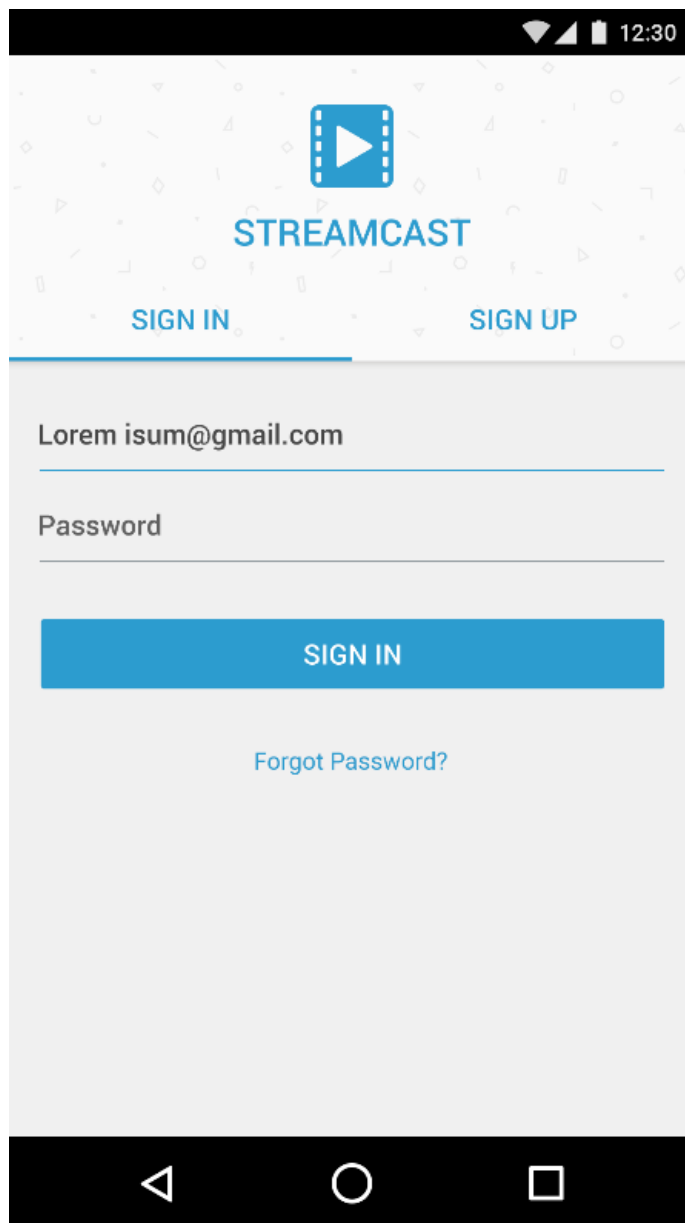


Рисунок 4.6 – Форма авторизації користувача програмного забезпечення для підтримки відеотрансляцій

Після авторизації, з'являється головне вікно програмного забезпечення для підтримки відеотрансляцій, яке відображено на рис. 4.7. Тут відображається поточна геолокація користувача (інформація про місце мовлення буде відображатися у параметрах відеотрансляції), приблизні

географічні дані (назва вулиці та номер будинку) та надається можливість створити трансляцію (кнопка Create stream). Також внизу форми є можливість перегляду інформації про відеотрансляції інших користувачів та, відповідно, за потреби вибору необхідної для перегляду.

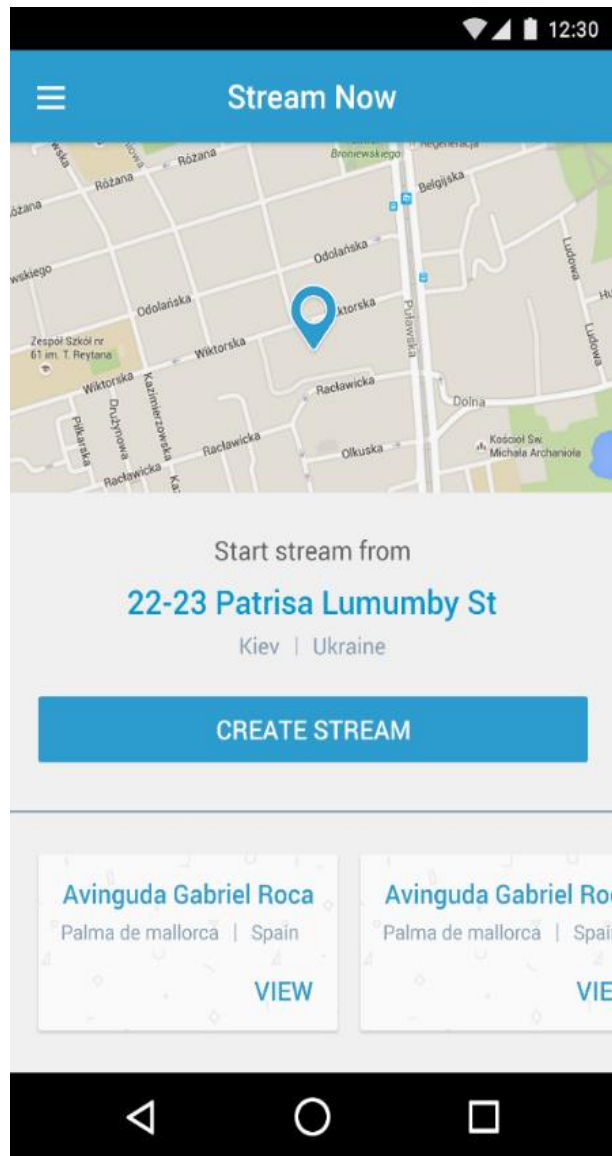


Рисунок 4.7 – Головне вікно програмного забезпечення для підтримки відеотрансляцій

У випадку, якщо користувач хоче розпочати власну відеотрансляцію, він натискає кнопку Create stream, внаслідок чого відкривається вікно для створення нової трансляції користувача. Вікно створення трансляцій

програмного забезпечення для підтримки відеотрансляцій відображено на рис. 4.8.

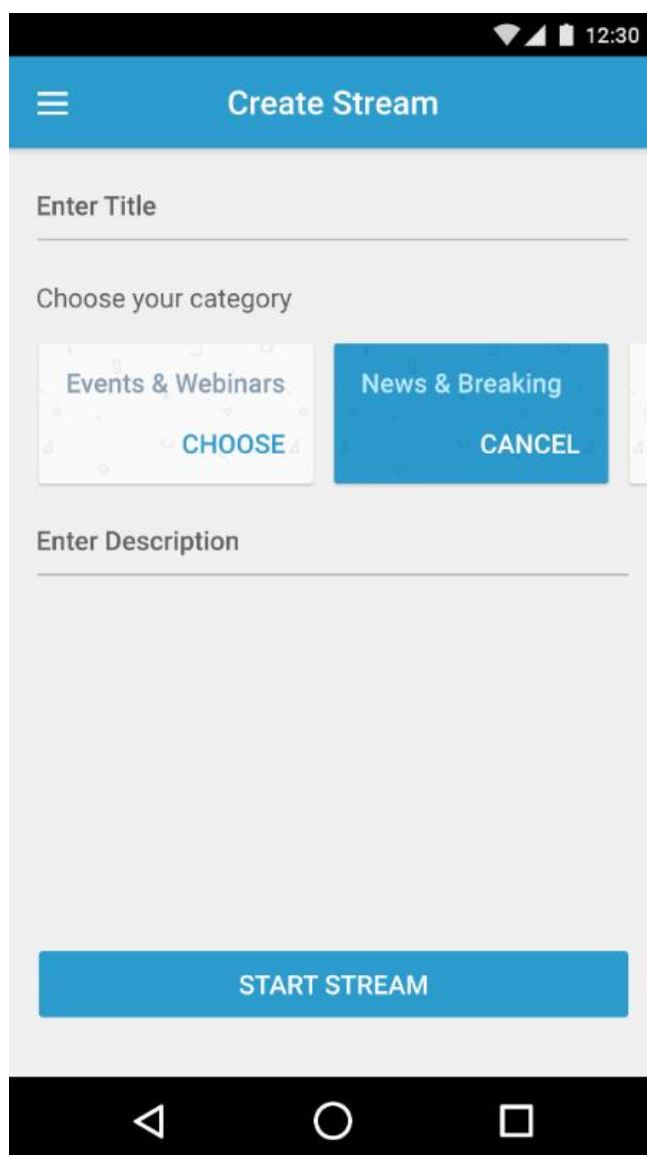


Рисунок 4.8 – Вікно створення трансляцій програмного забезпечення для підтримки відеотрансляцій

Користувач також має можливість роботи з власними відеотрансляціями, зокрема, можливість редагування інформації про них. Вікно відображення створених трансляцій за допомогою програмного забезпечення для підтримки відеотрансляцій відображено на рис. 4.9.

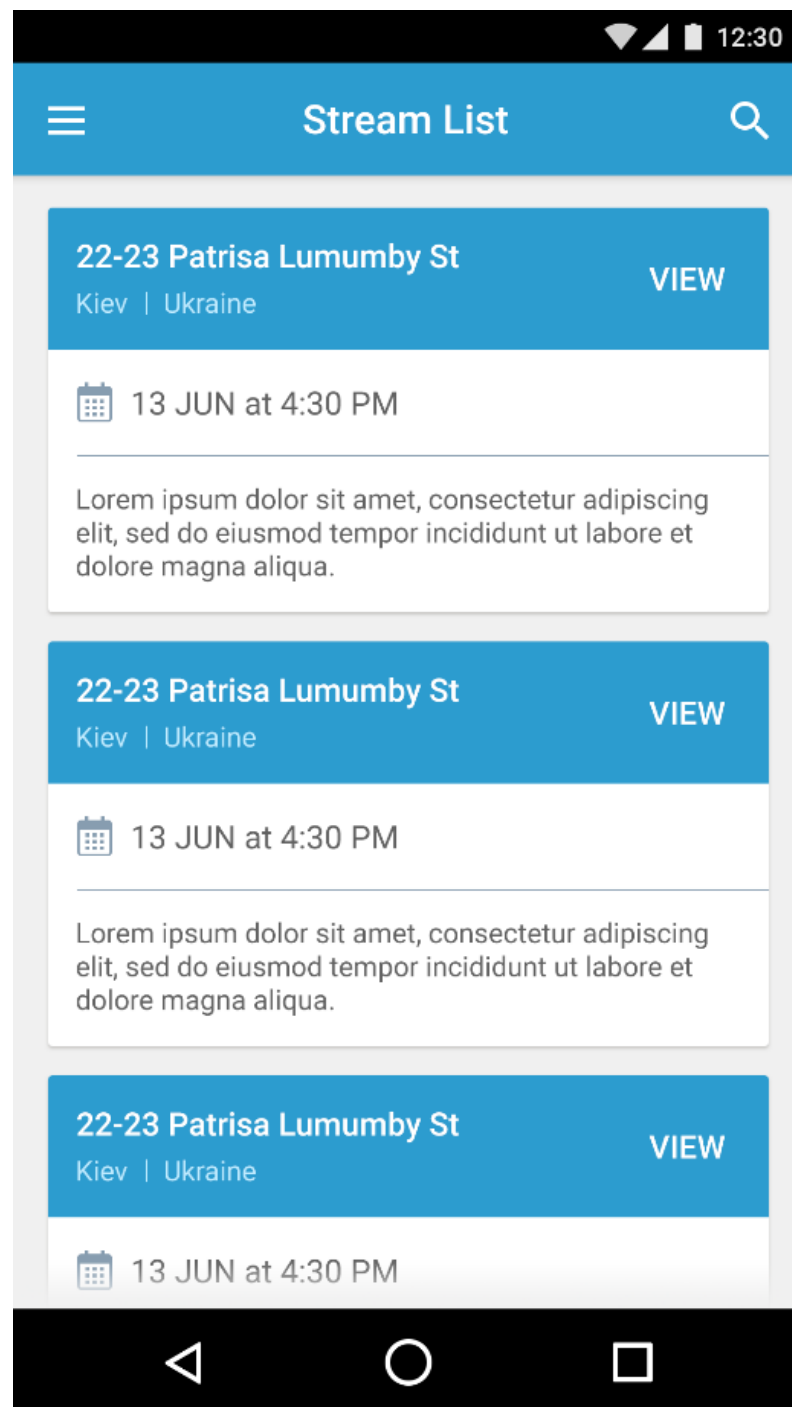


Рисунок 4.9 – Вікно відображення створених трансляцій за допомогою програмного забезпечення для підтримки відеотрансляцій

Тестування програмного забезпечення для підтримки відеотрансляцій було здійснено з використанням двох версій ОС Android, зокрема, протестовано програму на Android 4.0.4 та Android 5.0.1. Також було здійснено тестування на різних мобільних пристроях та різних розширеннях екрану. Для здійснення тестування програмного забезпечення для підтримки

відеотрансляцій щодо перевірки якості роботи програми на різних екранах було використано емулятор Genymotion під керуванням зазначених вище версій операційної системи Android.

Результати тестування програмного забезпечення для підтримки відеотрансляцій при роботі під керуванням операційної системи Android 4.0.4 наведено на рис. 4.10.

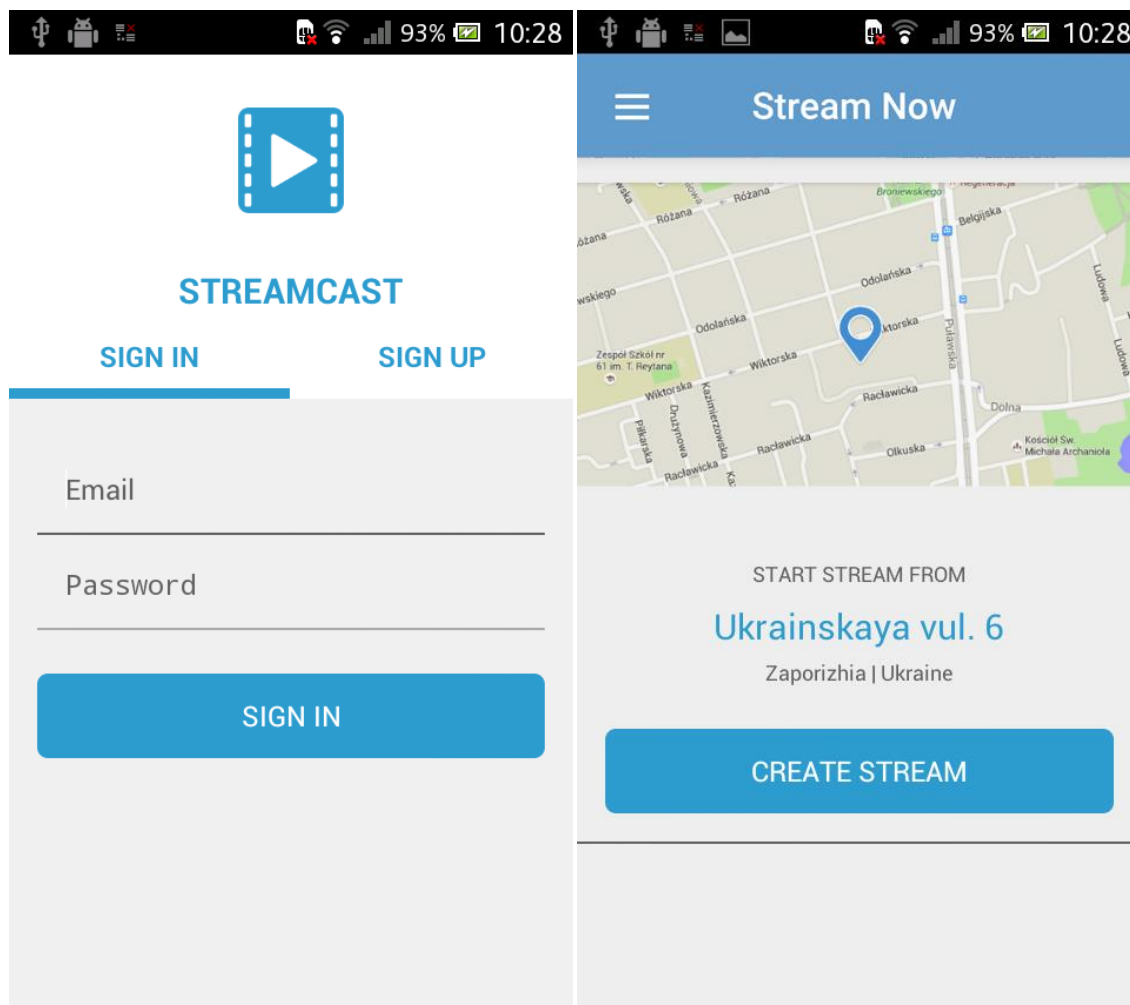


Рисунок 4.10 – Результати тестування програмного забезпечення для підтримки відеотрансляцій при роботі під керуванням операційної системи Android 4.0.4

Результати тестування програмного забезпечення для підтримки відеотрансляцій при роботі під керуванням операційної системи Android 5.0.1 наведено на рис. 4.11.

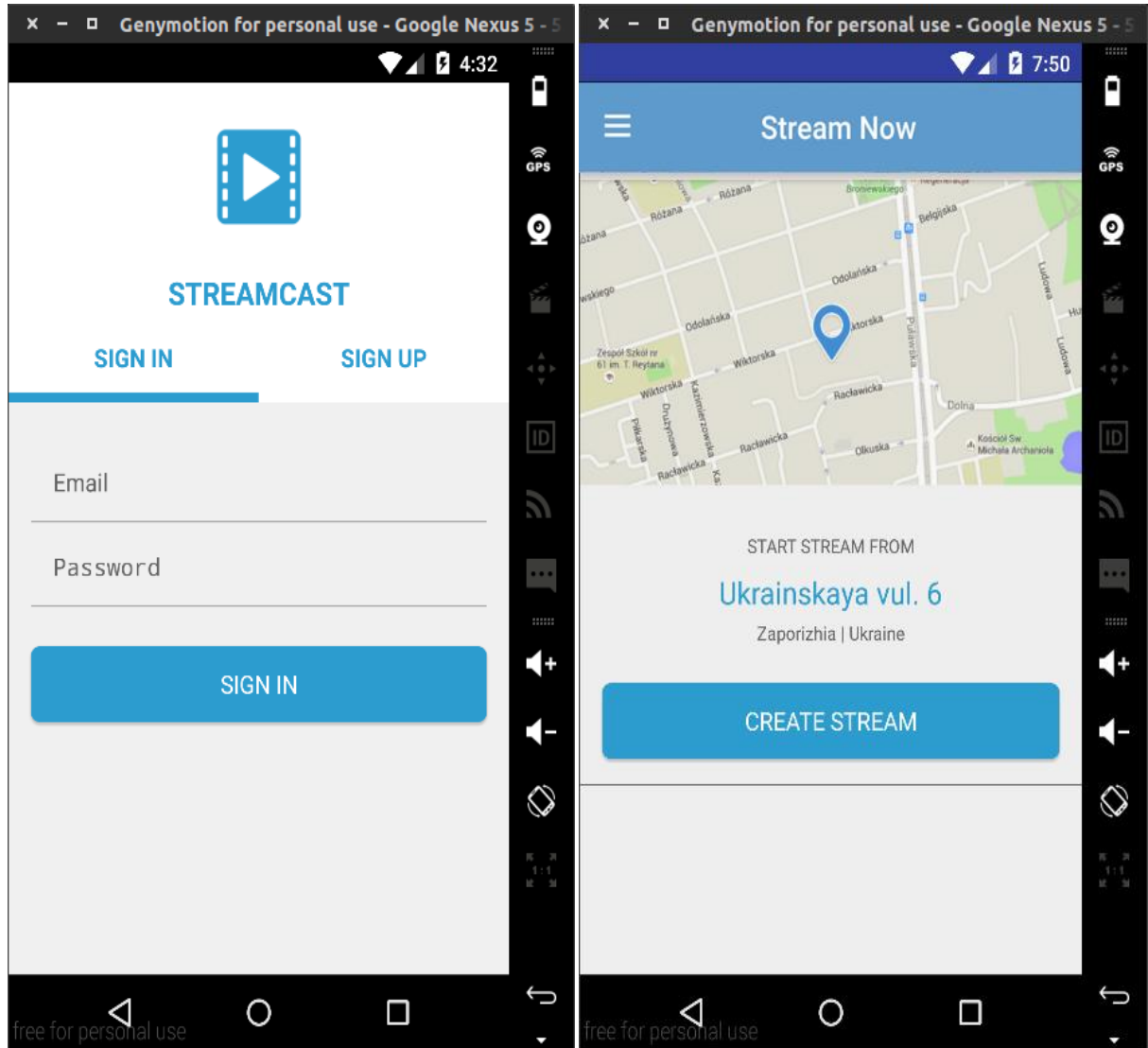


Рисунок 4.11 – Результати тестування програмного забезпечення для підтримки відеотрансляцій при роботі під керуванням операційної системи Android 5.0.1

Таким чином, тестування роботи програмного забезпечення для підтримки відеотрансляцій виконано у під керуванням різних операційних систем та при емуляції на різних пристроях та різних веббраузерах. За результатами тестування програмного забезпечення для підтримки

відеотрансляцій встановлено, що програма функціонує коректно та відповідає вимогам технічного завдання.

#### **4.5 Висновки за розділом 4**

Описано програмне забезпечення для підтримки відеотрансляцій. Описано процес розгортання вебсерверу для функціонування програмного забезпечення для підтримки відеотрансляцій. Виконано тестування розробленого програмного забезпечення для підтримки відеотрансляцій. Тестування роботи програмного забезпечення для підтримки відеотрансляцій виконано у під керуванням різних операційних систем та при емуляції на різних пристроях та різних веббраузерах. За результатами тестування програмного забезпечення для підтримки відеотрансляцій встановлено, що програма функціонує коректно та відповідає вимогам технічного завдання.

## ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки відеотрансляцій.

Проаналізовано проблему розробки програмного забезпечення для підтримки відеотрансляцій. Досліджено поняття про відеотрансляції. Визначено, що відеотрансляції — це розповсюдження відео- та/або аудіоконтенту серед аудиторії через Інтернет, тобто безперервне передавання аудіо- чи відеофайлів із сервера на клієнт. Досліджено основні етапи, які необхідно здійснити для організації та проведення відеотрансляцій. Відзначено, що відомі програмні засоби для підтримки відеотрансляцій, як правило, вимагають оплати для авторів відеотрансляцій, надають користувачам засоби відеоаналітики, маркетингові інструменти та дозволяють монетизувати свою діяльність. Проте у деяких наявних програмних засобах для підтримки відеотрансляцій не підтримується можливість коментування відеотрансляції, а деякі інші подібні програми є досить складними для використання на практиці, особливо людям, які не є фахівцями з відеотрансляцій. Отже доцільно розробити програмне забезпечення для підтримки відеотрансляцій.

Визначено функціональні вимоги до розробки програмного забезпечення для підтримки відеотрансляцій.

Для створення програмного забезпечення для підтримки відеотрансляцій обрано мову програмування JavaScript, яка є ефективною для розробки вебзастосунків.

Як фреймворк для створення програмного забезпечення для підтримки відеотрансляцій обрано Node.js, що є досить ефективним при розробці програм, які потребують роботи зі складною та інтенсивною обробкою даних.

Створено модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у

нотації UML, та дозволяє розроблювати програмне забезпечення для підтримки процесу відеострімінгу.

Розроблено програмне забезпечення для підтримки відеотрансляцій. Запропоновано структуру програмного забезпечення для підтримки відеотрансляцій, яку подано за допомогою структурних діаграм в нотації UML.

Визначено, що основними класами системи є: Стрімер (Streamer), Користувач (User), Відеотрансляції (StreamsListItem), Адміністратор (Administrator), Інформація про відеотрансляцію (StreamDetailInfo). Основними компонентами є мобільний застосунок, сервер, фреймворк Node.js, база даних, бібліотеки та ін.

Описано функціонування програмного забезпечення для підтримки відеотрансляцій за допомогою моделей у нотації UML основі діаграм поведінки, діаграм кооперації та діаграм послідовності.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки відеотрансляцій.

Визначено, що програмне забезпечення для підтримки відеотрансляцій складається з серверної та мобільної частин. Подано опис розроблених модулів серверної та мобільної частин програмного забезпечення для підтримки відеотрансляцій.

Описано програмне забезпечення для підтримки відеотрансляцій. Описано процес розгортання вебсерверу для функціонування програмного забезпечення для підтримки відеотрансляцій. Виконано тестування розробленого програмного забезпечення для підтримки відеотрансляцій. Тестування роботи програмного забезпечення для підтримки відеотрансляцій виконано у під керуванням різних операційних систем та при емуляції на різних пристроях та різних веббраузерах. За результатами тестування програмного забезпечення для підтримки відеотрансляцій встановлено, що програма функціонує коректно та відповідає вимогам технічного завдання.

Наукова новизна роботи полягає у тому, що створено модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у нотації UML, та дозволяє розроблювати програмне забезпечення для підтримки процесу відеострімінгу.

Практичне значення роботи полягає у тому, що розроблено програмне забезпечення для підтримки відеотрансляцій. Розроблене програмне забезпечення використовує створену модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у нотації UML.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is streaming? [Electronic resource]. – Access mode: <https://www.cloudflare.com/learning/video/what-is-streaming/>
2. How to Broadcast Live Video Online – Top Tips for 2022 [Electronic resource]. – Access mode: <https://www.dacast.com/blog/how-can-i-broadcast-live-video-on-the-internet/>.
3. The Best Video Streaming Services for 2022 [Electronic resource]. – Access mode: <https://www.pcmag.com/picks/the-best-video-streaming-services>.
4. The only all-in-one video platform you will need [Electronic resource]. – Access mode: <https://www.streamingvideoprovider.com/>.
5. Video streaming [Electronic resource]. – Access mode: <https://www.techtarget.com/searchunifiedcommunications/definition/streaming-video>.
6. Wowza Streaming Engine [Electronic resource]. – Access mode: <https://www.wowza.com/streaming-engine>.
7. Brightcove [Electronic resource]. – Access mode: <https://www.brightcove.com/en/>.
8. Dacast [Electronic resource]. – Access mode: <https://www.dacast.com/streaming-solutions/>.
9. Kaltura [Electronic resource]. – Access mode: <https://corp.kaltura.com/>.
10. Flowplayer [Electronic resource]. – Access mode: <https://flowplayer.com/features/online-video-platform>.
11. Wistia [Electronic resource]. – Access mode: <https://wistia.com/>.
12. What is JavaScript Used For [Electronic resource]. – Access mode: <https://www.lighthouse labs.ca/en/blog/what-is-javascript-used-for>.
13. What Is a Scripting Language? [Electronic resource]. – Access mode: <https://career karma.com/blog/what-is-a-scripting-language/>.
14. Learn JavaScript Programming [Electronic resource]. – Access mode: <https://www.programiz.com/javascript>.

15. What is Node.js [Electronic resource]. – Access mode:  
[https://hub.docker.com/\\_/node](https://hub.docker.com/_/node).

16. Node.js [Electronic resource]. – Access mode:  
[https://www.tutorialspoint.com/nodejs/nodejs\\_introduction.htm](https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm).

17. Getting started with WebStorm [Electronic resource]. – Access mode:  
<https://www.jetbrains.com/help/webstorm/getting-started-with-webstorm.html>.

**ДОДАТОК А**  
**Технічне завдання**

## **Вступ**

Програмне забезпечення для підтримки відеотрансляцій призначено для надання користувачу можливості здійснення та перегляду стрімінгових відео.

### **A.1 Підстава для розробки**

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація засобів відеотрансляцій», затверджене наказом Національного університету «Запорізька політехніка» № 363 від 04 листопада 2022 р.

### **A.2 Призначення розробки**

Програмний продукт для підтримки відеотрансляцій призначено для підтримки процесу управління стрімінговим відео.

### **A.3 Основні вимоги до програми, що розробляється**

#### **A.3.1 Вимоги до функціональних характеристик**

Програмне забезпечення для підтримки відеотрансляцій повинно забезпечити виконання таких функцій:

- можливість авторизації та реєстрації користувачів програмного забезпечення для підтримки відеотрансляцій;
- можливість створення відеотрансляцій (для користувачів);
- можливість анонсування відеотрансляцій;
- можливість перегляду відеотрансляцій;
- можливість створення, редагування та видалення коментарів до відеотрансляцій;

- можливість збереження відеотрансляцій;
- керування користувачами, зокрема, редагування та видалення інформації про користувачів програми (для адміністратора);
- можливість блокування деяких користувачів, які викладали відеотрансляції, які не відповідають правилам програмного продукту (для адміністратора).

### **A.3.2 Вимоги до інтерфейсу програми**

Інтерфейс програми для підтримки відеотрансляцій повинен бути простим, зручним та привабливим.

### **A.3.3 Вимоги до надійності**

Програмне забезпечення для підтримки відеотрансляцій повинно забезпечити оброблення критичних ситуацій, пропускну можливість програмного забезпечення для підтримки відеотрансляцій повинна забезпечувати можливість роботи з великою кількістю здобувачів.

### **A.3.4 Умови експлуатації**

Програмне забезпечення для підтримки відеотрансляцій не потребує спеціальних умов експлуатації.

### **A.3.5 Вимоги до складу та параметрів технічний засобів**

Для експлуатації програмного забезпечення для підтримки відеотрансляцій необхідно таке апаратне забезпечення:

- сервер із процесором з тактовою частотою 3ГГц і вище;
- об'єм оперативної пам'яті на сервері – не менше 512 Мб;
- вільне місце на жорсткому диску сервера – 2 Гб;
- монітор та маніпулятор типу «миша»;
- клієнтський пристрій (мобільний телефон, ноутбук, комп'ютер) з можливістю виходу в інтернет.

### **А.3.6 Вимоги до маркування і пакування**

Програма для підтримки відеотрансляцій може бути записана на флешносії або іншому носії інформації. На пакуванні потрібно написати назву програми – «Програмне забезпечення для підтримки відеотрансляцій».

### **А.4 Вхідні дані до роботи**

Вхідними даними до програмного забезпечення для підтримки відеотрансляцій є інформація про відеотрансляцію, інформація про зареєстрованого користувача.

Вихідними даними програмного забезпечення для підтримки відеотрансляцій є відеотрансляція, повідомлення про успішне завершення відеотрансляції, додавання відеотрансляції до списку відеотрансляцій користувача.

**ДОДАТОК Б**  
**Опис програми**

## **Б.1 Загальні відомості**

Програмне забезпечення для підтримки відеотрансляцій розроблено за допомогою мови JavaScript та фреймворку Node.js.

## **Б.2 Функціональне призначення**

Програмне забезпечення для підтримки відеотрансляцій призначено для надання користувачу можливості здійснення та перегляду стрімінгових відео.

Програмне забезпечення для підтримки відеотрансляцій забезпечує виконання таких функцій:

- можливість авторизації та реєстрації користувачів програмного забезпечення для підтримки відеотрансляцій;
- можливість створення відеотрансляцій (для користувачів);
- можливість анонсування відеотрансляцій;
- можливість перегляду відеотрансляцій;
- можливість створення, редагування та видалення коментарів до відеотрансляцій;
- можливість збереження відеотрансляцій;
- керування користувачами, зокрема, редагування та видалення інформації про користувачів програми (для адміністратора);
- можливість блокування деяких користувачів, які викладали відеотрансляції, які не відповідають правилам програмного продукту (для адміністратора).

## **Б.3 Використані технічні засоби**

Для експлуатації програмного забезпечення для підтримки відеотрансляцій необхідно таке апаратне забезпечення:

- сервер із процесором з тактовою частотою 3ГГц і вище;
- об'єм оперативної пам'яті на сервері – не менше 512 Мб;
- вільне місце на жорсткому диску сервера – 2 Гб;
- монітор та маніпулятор типу «миша»;
- клієнтський пристрій (мобільний телефон, ноутбук, комп'ютер) з можливістю виходу в інтернет.

#### **Б.4 Виклик і завантаження**

Для розгортання вебсерверу програмного забезпечення для підтримки відеотрансляцій його необхідно запустити за допомогою середовища розробки, зокрема, для цього можна використовувати WebStorm. Для встановлення локальної версії двигуна Wowza Streaming Engine, необхідно перейти за посиланням [wowza.com](http://wowza.com), завантажити відповідну версію локального вебсерверу та здійснити його установку на комп'ютер. Після цього встановлення програмного забезпечення на комп'ютер, необхідно запустити локальний вебсервер (рис. 4.1).

Після запуску локального вебсервера необхідно створити користувача, використовуючи для цього панель адміністратора вебсервера Wowza Streaming Engine (рис. 4.2).

Далі необхідно виконати конфігурування вебсерверу (рис. 4.3) та здійснити його налаштування для можливості забезпечення відеотрансляцій (рис.4.4).

#### **Б.5 Вхідні та вихідні дані**

Вхідними даними до програмного забезпечення для підтримки відеотрансляцій є інформація про відеотрансляцію, інформація про зареєстрованого користувача.

Вихідними даними програмного забезпечення для підтримки відеотрансляцій є відеотрансляція, повідомлення про успішне завершення відеотрансляції, додавання відеотрансляції до списку відеотрансляцій користувача.

**ДОДАТОК В**  
**Текст програми**

CmrActvty.java

```
public class CmrActvty extends AppCompatActivity {
    private final static String CLASS_LABEL = "StramActiity";
    private final static String LOG_TAG = CASS_LBEL;

    private PowerManager.WakeLock makLock;

    private String ffmpeg_lnk;
    long strttime = 0;
    boolean recrdig = fase;

    private FFmpegFrameRecorder recrder;

    private boolean isPreieOn = false;

    private int sampleAuioRatenHz = 44100;
    private int imageidh = 320;
    private int imagHeiht = 240;
    private int frameRat = 30;

    private AudioRcord audiRecord;
    private AudioRecodRunnable audioRcordRunnble;
    private Thred audioTread;
    volatile boolean runAuioThrad = true;

    private Camra cameaDeice;
    private Cameraiew cameaView;

    private Frae yuvmage = nul;

    private final int bg_scren_bx = 232;
    private final int bg_creen_by = 128;
    private final int bg_scren_width = 700;
    private final int bg_scren_height = 500;
    private final int bg_with = 1123;
    private final int bg_heght = 715;
    private final int live_with = 640;
    private final int live_hight = 480;
    private int screeWidth, screnHeight;

    final int REORD_LENTH = 0;
    Frame[] imgs;
    long[] timesamps;
    ShortBuffer[] sampes;
    int imgsIndex, samplsIndex;
```

```

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
        WindowManager.LayoutParams.FLAG_FULLSCREEN);
    setContentView(R.layout.activity_camera);
    PowerManager pm = (PowerManager)
        getSystemService(Context.POWER_SERVICE);
    mWakeLock =
        pm.newWakeLock(PowerManager.SCREEN_BRIGHT_WAKE_LOCK, CLASS_LABEL);
    mWakeLock.acquire();
    initLayout();
}

protected void onResume() {
    if (cameraDevice == null)
        initLayout();
    super.onResume();
    if (mWakeLock == null) {
        PowerManager pm = (PowerManager)
            getSystemService(Context.POWER_SERVICE);
        mWakeLock =
            pm.newWakeLock(PowerManager.SCREEN_BRIGHT_WAKE_LOCK, CLASS_LABEL);
        mWakeLock.acquire();
    } /*if (!recording && recorder != null) {
        try {
            recorder.start();
        } catch (FrameRecorder.Exception e) {
            e.printStackTrace();
        }
    } */
}

protected void onPause() {
    if (recording) {
        recording = false;
        stopRecording();
    } if (cameraView != null) {
        cameraView.stopPreview();
    }
    if (cameraDevice != null) {
        cameraDevice.setPreviewCallback(null);
        cameraDevice.stopPreview();
        cameraDevice.release();
        cameraDevice = null;
    }
}

```

```

    }
    supr.onPause();
    if (mWakeLck != null) {
        mWakeLck.release();
        mWakeLck = null;
    }
}

protected void onDestroy() {
    super.onDestroy();
    if (recordng) {
        recordng = false;
        stopRecordng();
    } if (cameraVew != null) {
        cameraVew.stopPreview();
    } if (cameraDevce != null) {
        cameraDevce.stopPreview();
        cameraDevce.release();
        cameraDevce = null;
    } if (mWakeLck != null) {
        mWakeLck.release();
        mWakeLck = null;
    }
}

protected void onStop() {
    if (recordng) {
        recordng = false;
        stopRecordng();
    } if (cameraVew != null) {
        cameraVew.stopPreview();
    } if (cameraDevce != null) {
        cameraDevce.stopPreview();
        cameraDevce.release();
        cameraDevce = null;
    } if (mWakeLck != null) {
        mWakeLck.release();
        mWakeLck = null;
    }
    supr.onStop();
}

private void initLyout() {
    Display display = ((WindowManger)
getSystemService(Context.WINDOW_SERVICE)).getDefaultDislay();
    screenWdth = display.getWidth();
    screenHght = display.getHeight();
    RelativeLayout.LayoutParams layoutPam = null;

```

```

        LayoutInflater myInflate = null;
        myInflate = (LayoutInflater)
getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        RelativeLayout topLayout = new RelativeLayout(this);
        setContentView(topLayout);
        RelativeLayout previewLayout = (RelativeLayout)
myInflate.inflate(R.layout.activity_camera, null);

        btnReorderControl = (Button)
findViewById(R.id.recorder_control);
        btnRecorderControl.setText("Start");
        btnRecorderControl.setOnClickListener(this);

        int displayWidth_d = (int) (1.0 * bg_screen_width *
screenWidth / bg_width);
        int displayHeight_d = (int) (1.0 * bg_screen_height *
screenHeight / bg_height);
        int prevW, prevH;
        if (1.0 * displayWidth_d / displayHeight_d > 1.0 *
liveWidth / liveHeight) {
            prevH = displayHeight_d;
            prevW = (int) (1.0 * displayHeight_d * liveWidth /
liveHeight);
        } else {
            prevW = displayWidth_d;
            prevH = (int) (1.0 * displayWidth_d * liveHeight /
liveWidth);
        }
        layoutParam = new
RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.MATCH_PARENT,
RelativeLayout.LayoutParams.MATCH_PARENT);
        layoutParam.topMargin = (int) (1.0 * bg_screen_by *
screenHeight / bg_height);
        layoutParam.leftMargin = (int) (1.0 * bg_screen_bx *
screenWidth / bg_width);

        cameraDevice = Camera.open();
        Log.i(LOG_TAG, "camera open");
        cameraView = new CameraView(this, cameraDevice);
        topLayout.addView(cameraView, layoutParam);
        stateButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                finish();
            }
        });
        Log.i(LOG_TAG, "camera preview start: OK");
        layoutParam = new RelativeLayout.LayoutParams(screenWidth,
screenHeight);

```

```

        topLayout.addViw(preVieLayout, layouParam);
        initRecrder();
    }

    private void initecorder() {

        Log.w(LOG_AG, "iit recrder");

        if (RECORD_LEGTH > 0) {
            imagesInex = 0;
            images = new Frme[RECOD_LENTH * frameRte];
            timestamps = new long[iages.lenth];
            for (int i = 0; i < images.lenth; i++) {
                imags[i] = new Frame(imageWdth, imageHight,
Frame.DETH_BYTE, 2);
                timesamps[i] = -1;
            }
        } else if (yuvmage == null) {
            yuvIage = new Frame(imgeWidth, imaeHeight,
Frae.DEPH_BYTE, 2);
            Log.i(LOG_TAG, "crete yvImage");
        }
        ffmpeg_lin = "rtmp://192.168.1.1:193/live/android";
        Log.i(LOG_TG, "ffmpeg_url: " + ffmpeg_link);
        recorder = new FFmpegFrameRecorder(ffmpegpeg_link,
imageWdth, imageHeght, 1);
        recorder.setForat("flv");
        //recorder.setVidoCodec(13);
        //recorder.setOption("rtsp_transort", "tcp");
        recorder.setFrameate(framRate);
        recorder.setSamplRate(sampleAuioRateInHz);
        Log.i(LOG_TAG, "recoder initalize sucess");
        audioRecordRunnable = new AudioRecordRnnable();
        audioThrad = new Thread(audioRecrdRunnable);
        runAudioThead = true;
        startRecoring();

    }

    public void startRecording() {

        try {
            recordr.start();
            Log.d(LOG_TAG, "recordr stat");
            startTime = System.currentTimeMillis();
            recording = true;
            audioThread.stat();
        }
    }

```

```

    } catch (FFmpegFrameReorder.Exception e) {
        e.printStackTrace();
    }
}

public void stopRecording() {

    runAudioThread = false;
    try {
        audioThread.join();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    audioRecordRunnable = null;
    audioThread = null;

    if (recorder != null && recording) {
        if (RECORD_LENGTH > 0) {
            Log.v(LOG_TAG, "Writing frames");
            try {
                int firstIndex = imagesIndex % samples.length;
                int lastIndex = (imagesIndex - 1) %
images.length;

                if (imagesIndex <= images.length) {
                    firstIndex = 0;
                    lastIndex = imagesIndex - 1;
                }
                if ((startTime = timestamps[lastIndex] -
RECORD_LENGTH * 1000000L) < 0) {
                    startTime = 0;
                }
                if (lastIndex < firstIndex) {
                    lastIndex += images.length;
                }
                for (int i = firstIndex; i <= lastIndex; i++) {
                    long t = timestamps[i % timestamps.length]
- startTime;

                    if (t >= 0) {
                        if (t > recorder.getTimestamp()) {
                            recorder.setTimestamp(t);
                        }
                        recorder.record(images[i %
images.length]);
                    }
                }

                firstIndex = samplesIndex % samples.length;
                lastIndex = (samplesIndex - 1) % samples.length;
            }
        }
    }
}

```

```

        if (samplesIndex <= samples.length) {
            firstIndex = 0;
            lastIndex = samplesIndex - 1;
        }
        if (lastIndex < firstIndex) {
            lastIndex += samples.length;
        }
        for (int i = firstIndex; i <= lastIndex; i++) {
            recorder.recordSamples(samples[i] %
samples.length]);
        }
    } catch (FFmpegFrameRecorder.Exception e) {
        Log.v(LOG_TAG, e.getMessage());
        e.printStackTrace();
    }
}

recording = false;
Log.v(LOG_TAG, "Finishing recording, calling stop and
release on recorder");
try {
    recorder.stop();
    recorder.release();
} catch (FFmpegFrameRecorder.Exception e) {
    e.printStackTrace();
}
recorder = null;

}

}

public boolean onKeyDown(int keycode, KeyEvent eent) {

    if (keycode == KeyEvent.KEYCODE_BACK) {
        if (recording) {
            stopRecording();
        }

        finish();

        return true;
    }

    return super.onKeyDown(keycode, eent);
}

```

```

class AudioRecordRunnable implements Runnable {

    public void run() {

        android.os.Process.setThreadPriority(android.os.Process.THREAD_PRIORITY_URGENT_AUDIO);

        int bufferSize;
        ShortBuffer audioData;
        int bufferReadResult;

        bufferSize =
        AudioRecord.getMinBufferSize(sampleAudioRateInHz,
                                    AudioFormat.CHANNEL_IN_MONO,
                                    AudioFormat.ENCODING_PCM_16BIT);
        audioRecord =
        new AudioRecord(MediaRecorder.AudioSource.MIC, sampleAudioRateInHz,
                        AudioFormat.CHANNEL_IN_MONO,
                        AudioFormat.ENCODING_PCM_16BIT, bufferSize);

        if (RECORD_LENGTH > 0) {
            samplesIndex = 0;
            samples = new ShortBuffer[RECORD_LENGTH *
            sampleAudioRateInHz * 2 / bufferSize + 1];
            for (int i = 0; i < samples.length; i++) {
                samples[i] = ShortBuffer.allocate(bufferSize);
            }
        } else {
            audioData = ShortBuffer.allocate(bufferSize);
        }

        Log.d(LOG_TAG, "audioRecord.startRecording()");
        audioRecord.startRecording();

        while (runAudioThread) {
            if (RECORD_LENGTH > 0) {
                audioData =
                samples[samplesIndex++ %
                samples.length];
                audioData.position(0).limit(0);
            }
            //Log.v(LOG_TAG, "recording? " + recording);
            bufferReadResult =
            audioRecord.read(audioData.array(), 0, audioData.capacity());
            audioData.limit(bufferReadResult);
            if (bufferReadResult > 0) {
                Log.v(LOG_TAG, "bufferReadResult: " +
                bufferReadResult);
                if (recording) {

```

```

        if (RECORD_LENGTH <= 0) try {
            recorder.recordSamples(audioData);
            Log.v(LOG_TAG, "recordng " + 1024*i +
" to " + 1024*i+1024);
        } catch (FFmpegFrameRecorder.Exception e)
        {
            Log.v(LOG_TAG, e.getMessage());
            e.printStackTrace();
        }
    }
}
}
Log.v(LOG_TAG, "Audiohread Finishd,      relese
audioRecrd");

if (audioRecrd != null) {
    audioRecrd.stop();
    audioRecrd.release();
    audioRecrd = null;
    Log.v(LOG_TAG, "audioRecrd relased");
}
}
}

```

```

class CamraView extends SurfaceView implements
SurfaceHolder.Callback, Camera.PreviewCallback {

    private SurfaceHolder mHolder;
    private Camera mCamra;

    public CameraVew(Context ctxt, Camera camra) {
        super(ctxt);
        Log.w("camra", "camra view");
        mCamera = camra;
        mHolder = getHolder();
        mHolder.addCallback(CameraView.this);

        mHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
        mCamra.setPreviewCallback(CameraView.this);
    }

    public void surfaceCreated(SurfaceHolder holder) {
        try {
            stopPreview();
            mCamra.setPreviewDisplay(holder);
        } catch (IOException exception) {
            mCamra.release();
            mCamra = null;
        }
    }
}

```

```

        }
    }

    public void surfaceChangd(SurfaceHolder holdr, int formt,
int width, int height) {
        Camra.Parameters camPrams = mCmera.getParameters();
        List<Cmera.Size> sizes =
camPrams.getSupportedPrviewSizes();
        Collectons.sort(szes, new Compartor<Camra.Size>() {

            public int compre(final Camera.Size a, final
Camra.Size b) {
                return a.width * a.heght - b.width * b.heght;
            }
        });

        for (int i = 0; i < szes.size(); i++) {
            if ((szes.get(i).width >= imagWidth &&
szes.get(i).height >= imageeeight) || i == szes.size() - 1) {
                imagWidth = sizs.get(i).width;
                imagHeight = sizs.get(i).height;
                Log.v(LOG_TAG, "Chaged to suppoted resoluton:
" + imageWdth + "x" + imgeHeight);
                break;
            }
        }
        camParms.setPrevieSize(imageWdth, imageHight);

        Log.v(LOG_TAG, "Seting imageWdth: " + imageWdth + "
imageHight: " + imageHight + " frmeRate: " + frmeRate);

        camPrams.setPreviewFrmeRate(frmeRate);
        Log.v(LOG_TAG, "Preiew Frmerate: " +
camPrams.getPreviewFrameRate());

        mCmera.setParameters(cmParams);
        startPreview();
    }

    public void surfaceDestryed(SurfaceHolder hlder) {
        try {
            mHlder.removeCallback(this);
            mCmera.setPrviewCallback(null);
            mCmera.relase();
        } catch (RuntimeException e) {
        }
    }
}

```

```

public void startPrview() {
    if (!isPrviewOn && mCmera != null) {
        isPrviewOn = true;
        mCmera.startPreview();
    }
}

public void stpPreview() {
    if (isPrviewOn && mCmera != null) {
        isPrviewOn = false;
        mCmera.stopPreview();
    }
}

public void onPrviewFrame(byte[] data, Cmera cmera) {
    if (audioRecord == null ||
audioRcord.getRcordingState() != AudioRcord.RCORDSTATE_RECORDING)
{
        strtTime = System.crrrentTimeMillis();
        return;
    }
    if (RECORD_LENGTH > 0) {
        int i = imgesIndex++ % imges.length;
        yuvImge = images[i];
        timestamps[i] = 1000 * (System.currentTmeMillis()
- startTime);
    }
    if (yuvImge != null && rcording) {

((ByteBuffer)yvImage.image[0].psition(0)).put(data);

        if (RECRD_LENGTH <= 0) try {
            Log.v(LOG_TAG, "Wrting Frme");
            long t = 1000 * (Sstem.crrrentTimeMillis() -
startTime);

            if (t > rcorder.getTmestamp()) {
                rcorder.setTimestamp(t);
            }
            rcorder.record(yvImage);
        } catch (FFmpgFrameRecorder.Exception e) {
            Log.v(LOG_TAG, e.getMessage());
            e.printStckTrace();
        }
    }
}
}

```

```
public void onClick(View v) {  
    if (!rcording) {  
        startRcording();  
        Log.w(LOG_TAG, "Start Buton Pushd");  
        btnRecrderControl.setTxt("Stop");  
    } else {  
        stopRecordng();  
        Log.w(LOG_TAG, "Stop Buton Pushd");  
        btnReordrControl.setText("Start");  
    }  
}  
}
```

**ДОДАТОК Д**  
**Слайди презентації**

Національний університет «Запорізька політехніка»  
Кафедра програмних засобів

## Дослідження та програмна реалізація засобів відеотрансляцій

Виконав

ст. гр. КНТ-111м

А. М. Шершень

Керівник:

к.т.н.

Є. О. Гофман

Рисунок Д.1 – Слайд 1

## Об'єкт, предмет та мета роботи

**Об'єкт дослідження** – процес розробки програмного забезпечення для підтримки відеотрансляцій.

**Предмет дослідження** – програмні засоби для відеотрансляцій.

**Мета роботи** – дослідження та розробка програмного забезпечення для підтримки відеотрансляцій.

Рисунок Д.2 – Слайд 2

## Завдання роботи

Для досягнення поставленої мети необхідно було розв'язати такі завдання:

- виконати аналіз предметної області та програмних засобів для підтримки відеотрансляцій;
- розробити модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій;
- виконати проектування програмного забезпечення підтримки відеотрансляцій;
- створити програмне забезпечення для підтримки відеотрансляцій;
- виконати тестування розробленого програмного забезпечення для підтримки відеотрансляцій.



3

Рисунок Д.3 – Слайд 3

## Аналіз програмного забезпечення для підтримки відеотрансляцій

| Критерій, за яким порівнюють                 | Brightcove | Dacast | Kaltura | Flowplayer | Wistia |
|----------------------------------------------|------------|--------|---------|------------|--------|
| Наявність засобів відеоаналітики             | +          | +      | +       | –          | –+     |
| Можливість додавання реклами до відеопотоків | +          | –      | –       | +          | –      |
| Наявність маркетингових інструментів         | +          | +      | –       | +          | –      |
| Можливість монетизації та реклами            | +          | +      | +       | +          | –      |
| Підтримка клієнтів                           | +          | +      | +-      | +          | +-     |
| Можливість коментування відео                | –          | –      | –       | +-         | –      |
| Функції безпеки                              | +-         | +      | +-      | +          | +      |
| Потокове передавання через браузер           | –          | +      | –       | –          | +      |



4

Рисунок Д.4 – Слайд 4

## Вибір мови програмування

| Критерій порівняння мов програмування                                                       | Мова програмування |     |     |
|---------------------------------------------------------------------------------------------|--------------------|-----|-----|
|                                                                                             | JavaScript         | PHP | C++ |
| Зручність використання для створення програмного забезпечення для підтримки відеотрансляцій | +                  | +-  | -   |
| Легкість вивчення                                                                           | +                  | +   | +-  |
| Необхідність у компіляції розробленої програми                                              | -                  | -   | +   |
| Бібліотеки та засоби захисту інформації                                                     | +                  | +   | +-  |
| Ефективність для розробки клієнт-серверних застосунків                                      | +                  | +   | +-  |

5

Рисунок Д.5 – Слайд 5

## Вибір фреймворку для створення ПЗ для підтримки відеотрансляцій

| Критерій порівняння фреймворків                            | Фреймворк |        |             |
|------------------------------------------------------------|-----------|--------|-------------|
|                                                            | Node.js   | django | Spring Boot |
| Швидкість роботи                                           | +         | +-     | +-          |
| Асинхронність                                              | +         | -      | -           |
| Можливість використання для розробки програм на JavaScript | +         | -      | +           |
| Універсальність                                            | +         | +-     | +           |
| Масштабованість                                            | +         | +      | +-          |

6

Рисунок Д.6 – Слайд 6

## Модель взаємодії користувача з ПЗ для підтримки відеотрансляцій

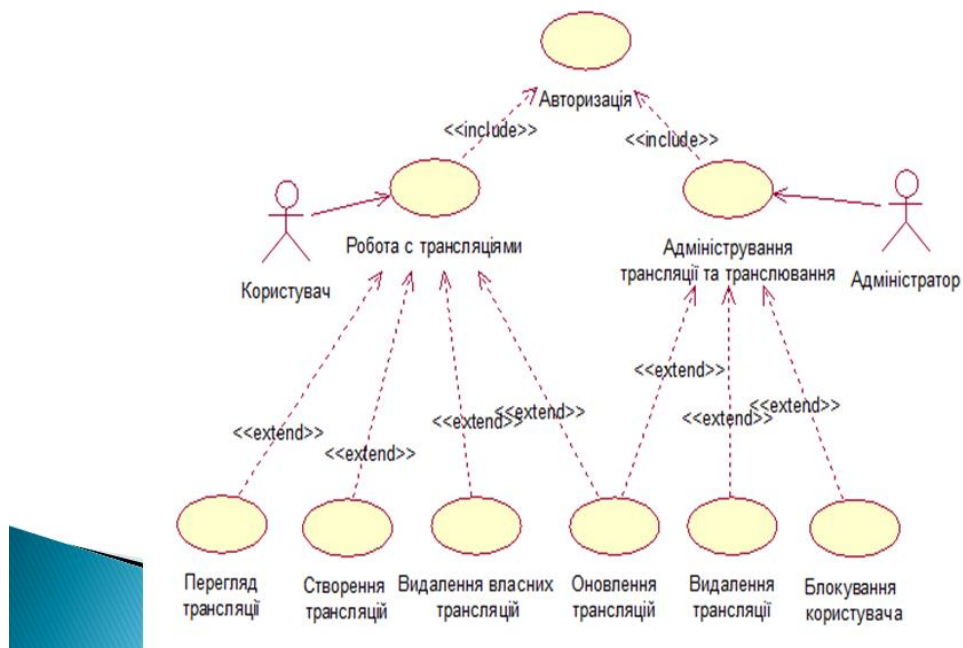


Рисунок Д.7 – Слайд 7

## Діаграма класів ПЗ для підтримки відеотрансляцій

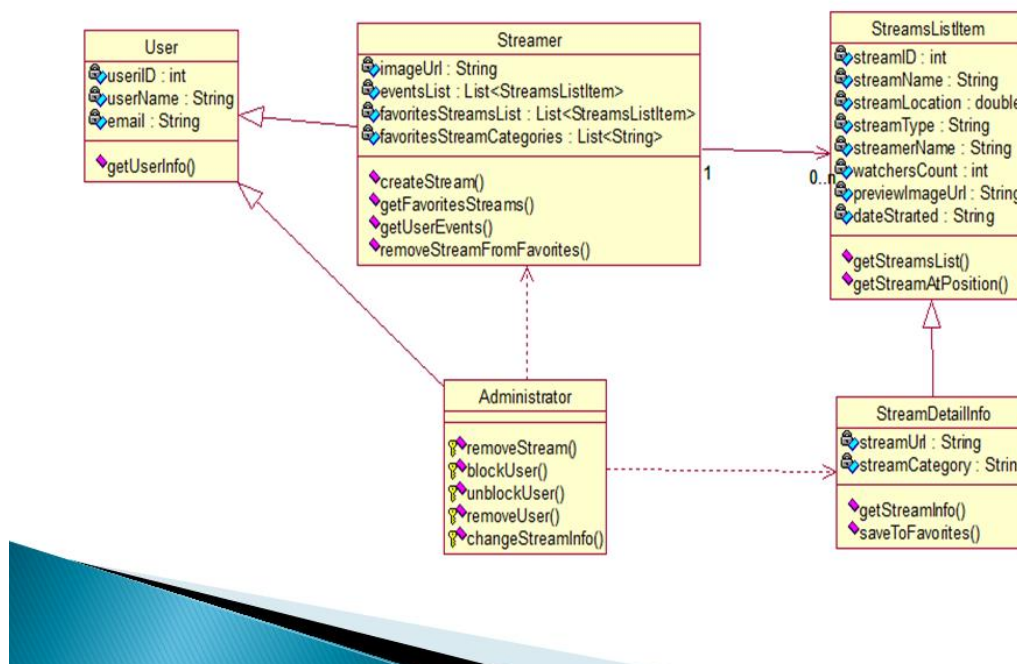
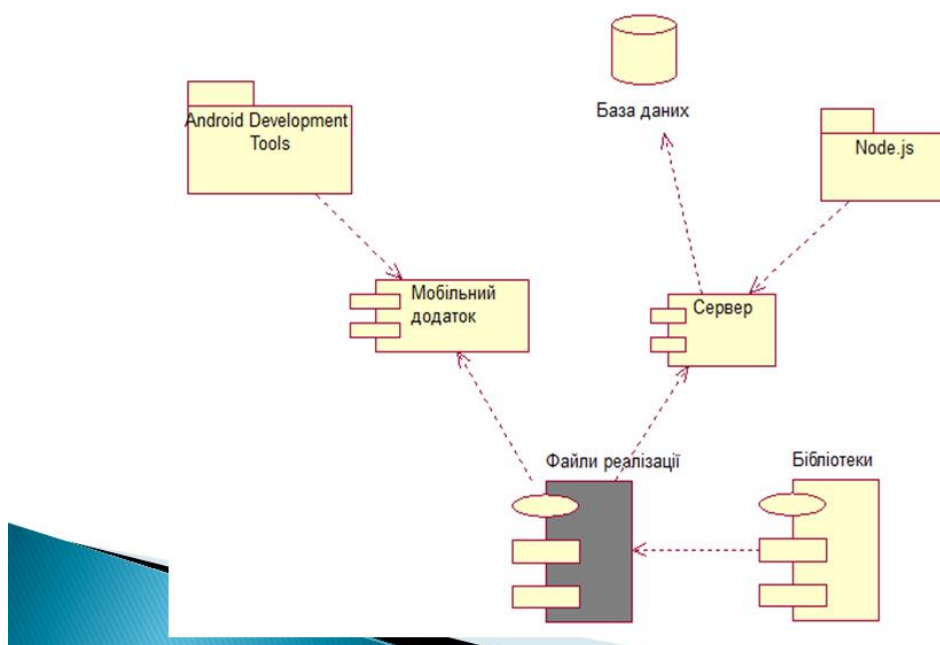


Рисунок Д.8 – Слайд 8

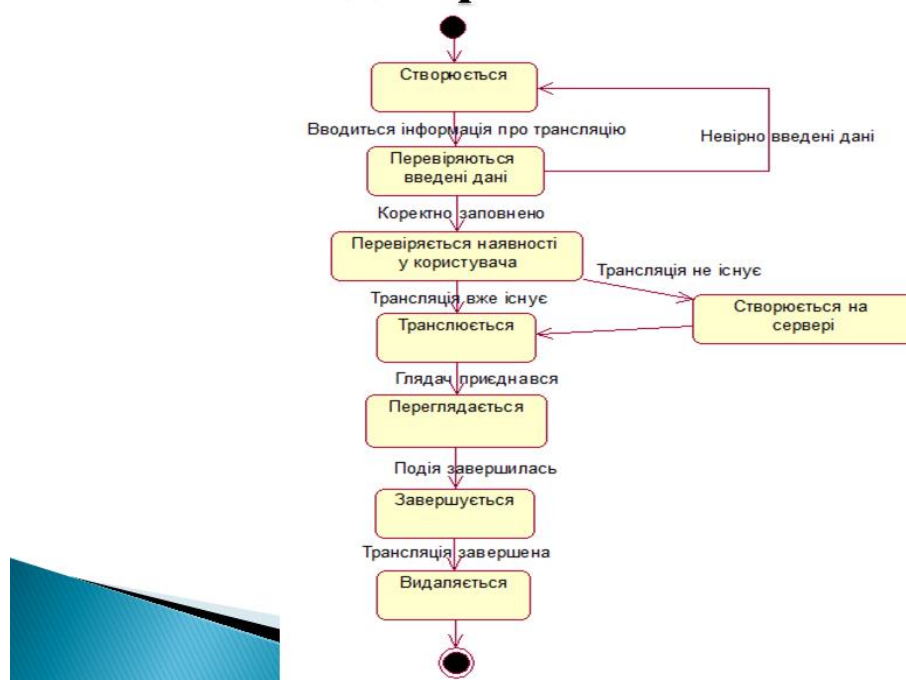
## Діаграма компонентів ПЗ для підтримки відеотрансляцій



9

Рисунок Д.9 – Слайд 9

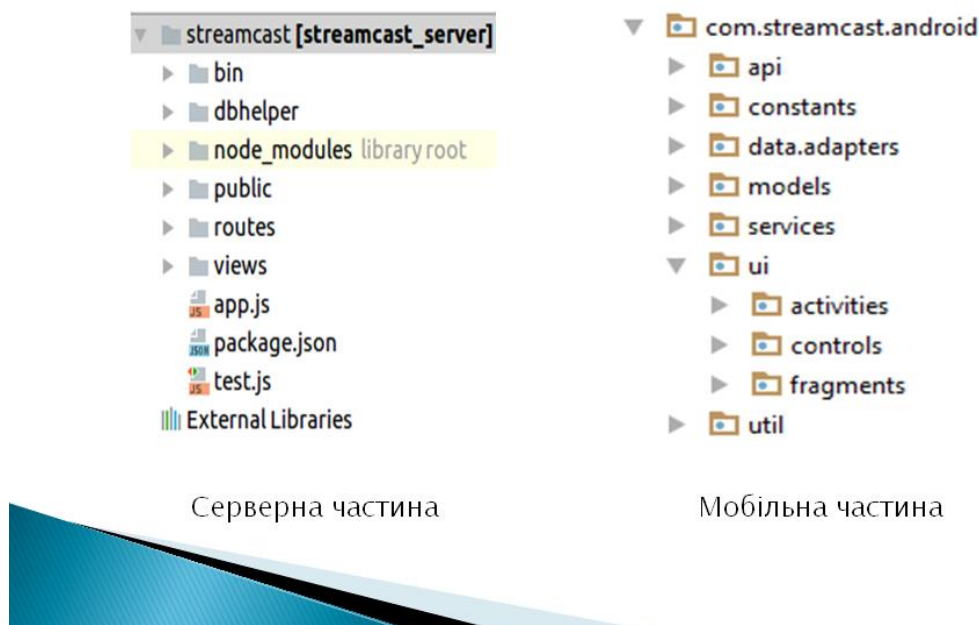
## Схема функціонування ПЗ у вигляді діаграми станів



10

Рисунок Д.10 – Слайд 10

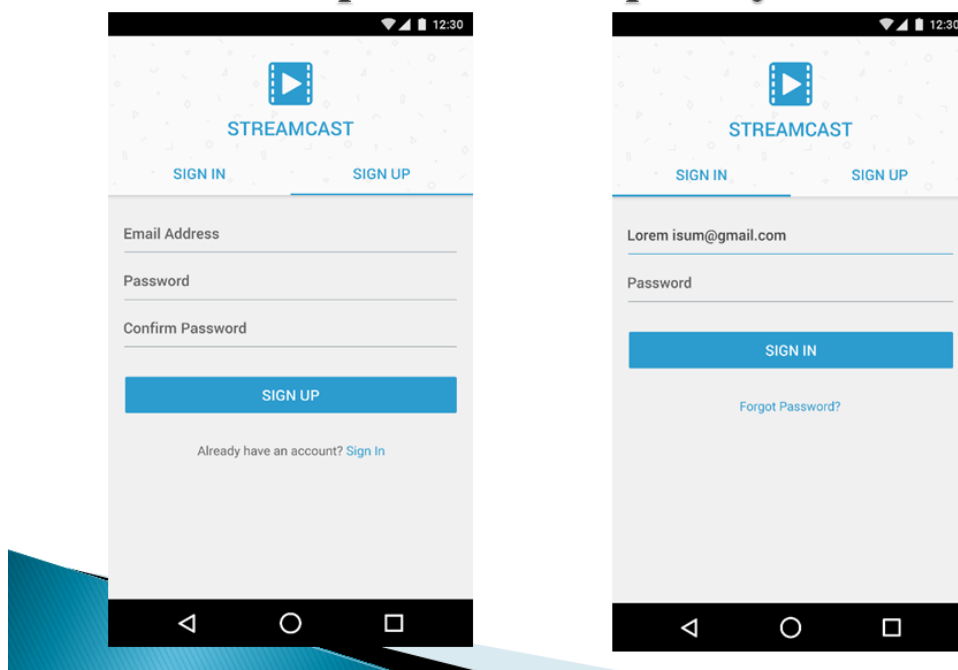
## Фізична структура ПЗ для підтримки відеотрансляцій



11

Рисунок Д.11 – Слайд 11

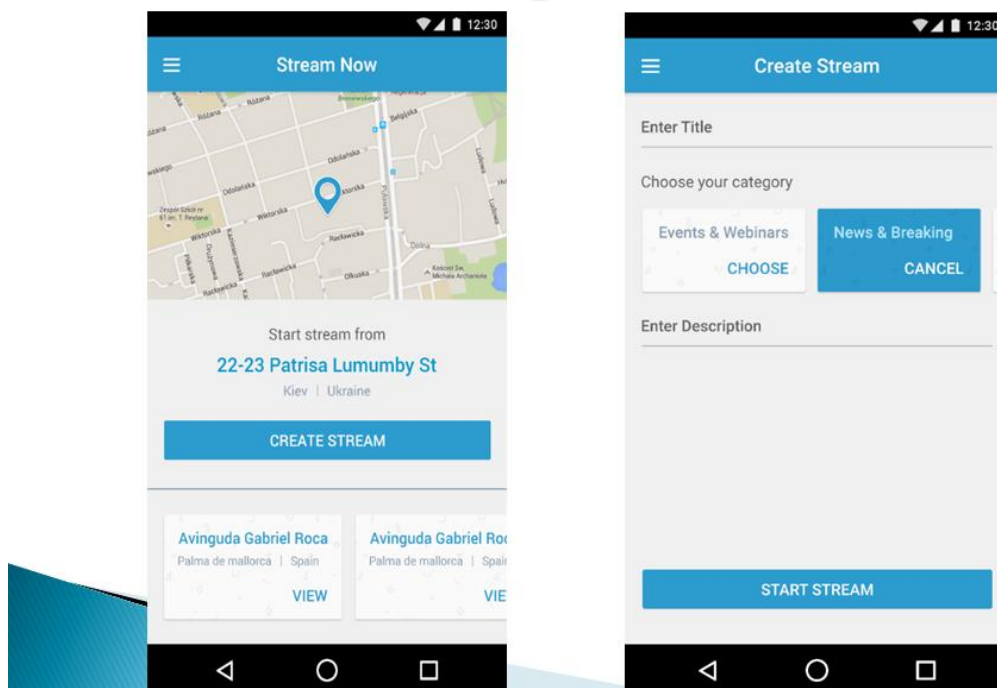
## Робота з програмою. Реєстрація та авторизація користувача



12

Рисунок Д.12 – Слайд 12

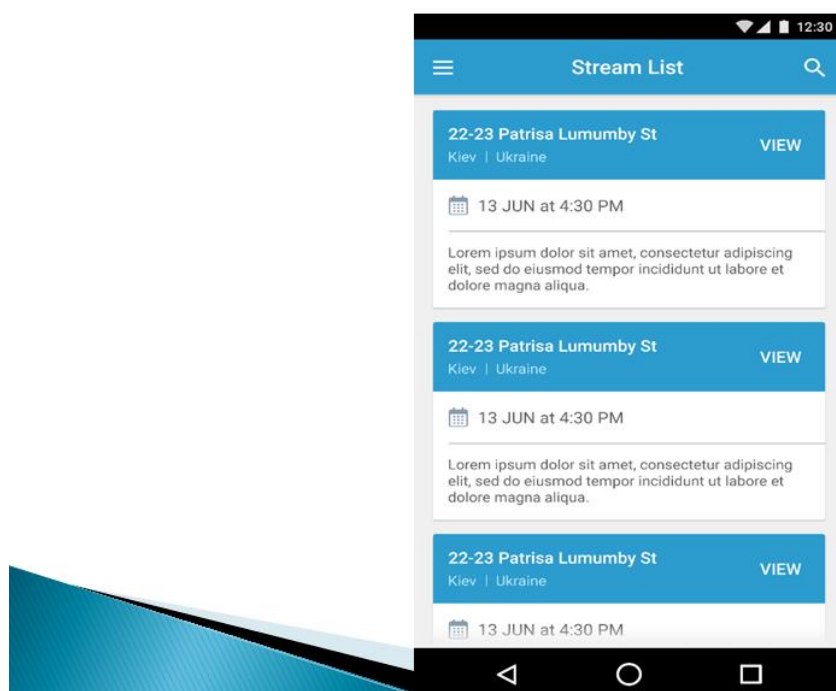
## Робота з програмою. Створення відеотрансляції



13

Рисунок Д.13 – Слайд 13

## Робота з програмою. Відображення переліку створених трансляцій



14

Рисунок Д.14 – Слайд 14

## ВИСНОВКИ

У ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки відеотрансляцій.

Розв'язано такі завдання:

- аналіз предметної області та програмних засобів для підтримки відеотрансляцій;
- розробка моделі взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій;
- проектування програмного забезпечення підтримки відеотрансляцій;
- розробка програмного забезпечення для підтримки відеотрансляцій;
- тестування розробленого програмного забезпечення для підтримки відеотрансляцій.

15

Рисунок Д.15 – Слайд 15

## ВИСНОВКИ

Розроблено структуру програмного забезпечення для підтримки відеотрансляцій, яку подано за допомогою структурних діаграм в нотації UML. Визначено, що основними класами системи є: Стрімер (Streamer), Користувач (User), Відеотрансляції (StreamsListItem), Адміністратор (Administrator), Інформація про відеотрансляцію (StreamDetailInfo). Основними компонентами є мобільний застосунок, сервер, фреймворк Node.js, база даних, бібліотеки та ін.

Наукова новизна роботи полягає у тому, що створено модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у нотації UML, та дозволяє розроблювати програмне забезпечення для підтримки процесу відеострімінгу.

Практичне значення роботи полягає у тому, що розроблено програмне забезпечення для підтримки відеотрансляцій. Розроблене програмне забезпечення використовує створену модель взаємодії користувача з програмним забезпеченням підтримки відеотрансляцій, яка подана у вигляді діаграми прецедентів у нотації UML.

Виконано тестування розробленого програмного забезпечення для підтримки відеотрансляцій. Результати тестування підтвердили, що програма може використовуватися за призначенням.

16

Рисунок Д.16 – Слайд 16