I N F O R M A T I K A
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

IMPLEMENTASI HTTP LIVE STREAMING (HLS) UNTUK OPTIMASI SISTEM CCTV PEMERINTAH KOTA PEKANBARU

Dedy Yasriady[1*], Yulisman[2], Muhammad Fais Avriody Daffa[3]
[1]Universitas Awal Bros Pekanbaru
[2,3]Universitas Hang Tuah Pekanbaru
[1]Jl. Karyabakti Jl. Bambu Kuning No.8, Rejosari, Kec. Tenayan Raya, Kota Pekanbaru, Riau 28141.
[2,3]Jl. Mustafa Sari No. 5, Kel. Tangkerang Selatan, Kec. Bukit Raya, Pekanbaru, Riau 28288.
E-mail : yasriady@gmail.com

ABSTRAK

Penelitian ini bertujuan mengoptimalkan sistem CCTV Pemerintah Kota Pekanbaru guna meningkatkan kemudahan akses publik. Permasalahan utama adalah keterbatasan akses streaming yang membutuhkan plugin atau aplikasi tambahan. Solusi yang ditawarkan adalah implementasi teknologi HTTP Live Streaming (HLS), dengan integrasi RTSP sebagai sumber video, FFmpeg sebagai transcoder, dan HLS sebagai protokol distribusi berbasis HTTP. Metode penelitian menggunakan pendekatan eksperimental melalui penerapan sistem HLS pada infrastruktur CCTV yang sudah ada, diikuti pengujian pada berbagai browser umum serta evaluasi performa sistem melalui pengukuran konsumsi CPU, trafik jaringan, dan stabilitas streaming. Hasil monitoring menunjukkan bahwa sistem mampu menampilkan video secara stabil di browser populer seperti Chrome, Firefox, Safari, dan Edge. Meskipun konsumsi CPU dan bandwidth cukup tinggi, sistem ini secara signifikan meningkatkan aksesibilitas layanan CCTV karena dapat diakses langsung melalui browser. HLS terbukti menjadi solusi efektif dan potensial dalam mendukung pengembangan sistem smart city di Indonesia.

Kata kunci: *hls, streaming, cctv, smart city, pekanbaru*;

ABSTRACT

This study aims to optimize the CCTV system of the Pekanbaru City Government to enhance public accessibility. The main issue addressed is the limited access to streaming, which typically requires third-party plugins or applications. The proposed solution involves implementing HTTP Live Streaming (HLS) technology by integrating RTSP as the video source, FFmpeg as the transcoder, and HLS as the HTTP-based distribution protocol. The research method uses an experimental approach by deploying the HLS system on the existing CCTV infrastructure, followed by testing on various mainstream browsers and performance evaluation through measurements of CPU usage, network traffic, and streaming stability. Monitoring results show that the system can display video stably across popular browsers such as Chrome, Firefox, Safari, and Edge. Although CPU and bandwidth consumption is relatively high, the system significantly improves public CCTV accessibility as users can access it directly via browser. HLS proves to be an effective and promising solution to support smart city development in Indonesia.

Keywords: *hls, streaming, cctv, smart city, pekanbaru;*

I N F O R M A T I K A
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

1. PENDAHULUAN

Di era digital, konsep smart city menempatkan sistem keamanan publik—seperti CCTV—sebagai elemen krusial untuk menciptakan deteksi dini terhadap tindak kriminal dan meningkatkan partisipasi warga. Studi kasus Kota Bandung menunjukkan bahwa penerapan smart *surveillance* melalui jaringan kamera publik telah efektif dalam mendukung ketertiban sosial dan menjaga keamanan perkotaan. Namun keberhasilannya sangat dipengaruhi oleh faktor-faktor seperti integrasi teknologi, regulasi yang mendukung, dan literasi masyarakat yang cukup (Alkaf & Sutrisno, 2019). Meskipun teknologi sistem pengawasan seperti CCTV telah diimplementasikan di banyak kota, keefektivitasannya tidak semata-mata bergantung pada perangkat dan integrasi teknologi. Keberhasilan sistem ini juga sangat ditentukan oleh kualitas kebijakan yang mengaturnya, serta partisipasi aktif masyarakat—yang disebut sebagai unsur smart people—dalam penggunaan dan pengawasan sistem tersebut. Studi sistematis telah menunjukkan bahwa keamanan perkotaan yang didukung teknologi (*smart city security*) memerlukan keterlibatan masyarakat untuk menjaga legitimasi dan efektivitas pengawasan publik (Laufs et al., 2020). Dukungan terhadap partisipasi masyarakat tidak hanya meningkatkan transparansi dan akuntabilitas, tetapi juga memperkuat rasa aman kolektif melalui co-creation tata kelola kota (KUMAR, 2024).Jika diadaptasikan pada konteks Kota Pekanbaru, pengembangan sistem CCTV publik dapat diarahkan untuk memperkuat keamanan lingkungan sekaligus membuka akses pemantauan yang lebih transparan bagi warga. Implementasi CCTV yang efektif dapat meningkatkan deteksi dini tindak kriminal dan menciptakan rasa aman kolektif, selama didukung oleh akses yang merata dan regulasi perlindungan data yang memadai (Yang et al., 2024). Hasil ini menggarisbawahi pentingnya integrasi antara teknologi dan partisipasi publik dalam pengembangan sistem keamanan kota berbasis digital.

Streaming tradisional umumnya menggunakan protokol *stateful*, misalnya, *Real-Time Streaming Protocol* (RTSP): Setelah klien terhubung ke server *streaming*, server akan terus memantau status klien hingga klien terputus lagi. Sebaliknya, HTTP bersifat *stateless*, setiap permintaan HTTP ditangani sebagai transaksi satu kali yang sepenuhnya mandiri(Stockhammer, 2011). Namun, Salah satu tantangan utama dalam sistem pemantauan video secara daring adalah variabilitas kompatibilitas pada sisi pengguna akhir, terutama saat mengakses tayangan melalui browser modern. Studi menunjukkan bahwa dukungan pemutaran video dapat berbeda-beda antara browser seperti Chrome, Firefox, Edge, dan Safari, sehingga menuntut pendekatan adaptif dan strategi optimasi yang matang untuk menjaga kualitas pengalaman pengguna (Yang et al., 2024). Teknologi streaming konvensional seperti RTSP (*Real-Time Streaming Protocol*) umumnya tidak dapat diputar langsung melalui browser tanpa bantuan plugin tambahan atau software pihak ketiga. Hal ini menyebabkan terbatasnya aksesibilitas bagi masyarakat umum yang menggunakan perangkat standar tanpa konfigurasi khusus.

Sebagai solusi atas permasalahan tersebut, protokol HTTP Live Streaming (HLS), yang dikembangkan oleh Apple Inc., membagi aliran video menjadi segmen-segmen kecil berformat .ts dan mengelolanya dengan playlist .m3u8, lalu disalurkan melalui HTTP standar. Pendekatan ini dirancang untuk adaptasi dinamis terhadap kondisi jaringan dan menjamin pengiriman konten multimedia yang stabil dan andal (Saini & Sharma, 2024). Teknologi ini tidak hanya mendukung akses lintas *platform* (*desktop* dan *mobile*), tetapi juga menawarkan toleransi jaringan yang lebih baik dan kemudahan integrasi dengan web server biasa seperti Nginx atau Apache. HTTP Live Streaming (HLS), yang dikembangkan oleh Apple Inc. sejak 2009, membagi konten video menjadi segmen (.ts) yang dikontrol melalui playlist .m3u8 dan disajikan via HTTP, sehingga mampu mendukung *adaptive bitrate*, lintas platform, serta optimalisasi kualitas di gradasi jaringan—semua tanpa memerlukan plugin khusus dan memungkinkan integrasi simpel dengan server HTTP seperti Nginx atau Apache (Saini & Sharma, 2024; Pantos & Apple Inc., 2025; MDPI, 2023). Selain itu, Pengiriman berbasis HTTP memungkinkan layanan streaming yang mudah dan tanpa usaha dengan menghindari masalah traversal NAT dan firewall(Stockhammer, 2011).

Studi terbaru oleh Nunez & Toasa, (2020) mengevaluasi performa protokol RTMP, RTSP, dan HLS pada jaringan mobile dengan kondisi jaringan yang beragam. Hasil penelitian mereka menunjukkan bahwa HLS merupakan protokol yang paling unggul dalam hal stabilitas, terutama saat terjadi perubahan kecepatan internet, serta menawarkan konsumsi baterai dan sumber daya yang seimbang.

I N F O R M A T I K A
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

Kemampuan HLS untuk membagi video menjadi segmen-segmen kecil dan beradaptasi dengan bitrate secara dinamis membuatnya ideal untuk lingkungan dengan fluktuasi jaringan, seperti yang ditemui di Kota Guaranda, Ecuador.

Kota Pekanbaru telah mengalami ekspansi pesat terhadap infrastruktur bangunan dalam dua dekade terakhir, dimana built-up area kota ini meningkat sekitar 13,45 % antara tahun 2000 hingga 2020, dengan rata-rata pertumbuhan tahunan mencapai 0,67 % (Almegi & Fatmawati, 2024), membutuhkan solusi streaming yang dapat mengatasi variasi jaringan dan memastikan pemantauan yang lancar tanpa interupsi. Dengan mengadopsi HLS, sistem CCTV dapat mencapai stabilitas tinggi dan mengoptimalkan beban sumber daya pada perangkat pengguna. Studi menunjukkan bahwa pemutaran video via jaringan 4G LTE mengonsumsi energi secara signifikan—karena pola transmisi TCP burst—namun HLS yang memanfaatkan segmentasi .ts dan kontrol adaptif bitrate secara objektif mampu menurunkan konsumsi daya perangkat hingga 40–70% dibanding metode stream lain (Zhang et al., 2018). Hal ini sangat relevan untuk memastikan efisiensi daya pada perangkat pengguna, baik saat menggunakan jaringan kabel, nirkabel, 4G, maupun 3G. Selain itu, implementasi HLS juga sejalan dengan kebutuhan akan skalabilitas dan keamanan, mengingat protokol ini berbasis HTTP yang mudah diintegrasikan dengan infrastruktur existing dan mendukung enkripsi data. Teknologi HLS berbasis HTTP dirancang untuk mudah disesuaikan dengan infrastruktur yang sudah ada seperti Nginx atau Apache, serta menyediakan toleransi jaringan yang baik. Karena semua distribusi konten berjalan melalui protokol HTTP standar, HLS dapat menembus *firewall* atau *proxy* secara otomatis dan memanfaatkan cache HTTP umum dalam skala besar. Selain itu, protokol ini menyertakan mekanisme enkripsi serta distribusi kunci melalui HTTPS, memungkinkan implementasi sistem DRM sederhana untuk meningkatkan keamanan data (Seufert et al., 2015). Melalui penelitian ini, kontribusi utama yang diharapkan adalah peningkatan kualitas layanan CCTV publik, yang pada akhirnya mendukung efektivitas pengawasan dan keamanan di Kota Pekanbaru.

Dalam implementasinya, sistem HLS dapat digabungkan dengan perangkat lunak *transcoder* seperti *FFmpeg* dan layanan manajemen proses seperti *supervisord* untuk mengelola banyak kamera secara paralel. FFmpeg (n.d.) mendukung fungsi *transcoding* untuk berbagai protokol *streaming*, termasuk konversi dari RTSP (umum digunakan dalam sistem CCTV) ke HLS (HTTP Live Streaming), yang memungkinkan distribusi video yang lebih efisien melalui jaringan berbasis HTTP. (Thanh Le et al., 2019)

Penelitian ini bertujuan untuk mengevaluasi implementasi HLS dalam sistem pemantauan CCTV Kota Pekanbaru, dengan fokus pada **aspek kompatibilitas pengguna**, **performa, efisiensi sumber daya,** dan diharapkan hasil penelitian ini dapat menjadi acuan bagi pengembangan sistem pemantauan serupa di daerah lain dalam rangka mendukung transformasi digital menuju kota cerdas (*smart city*).

Implementasi protokol HTTP Live Streaming (HLS) pada sistem CCTV Pemerintah Kota Pekanbaru memberikan sejumlah manfaat penting, mulai dari peningkatan aksesibilitas publik tanpa memerlukan aplikasi tambahan, kompatibilitas lintas platform melalui HLS.js, hingga efisiensi konsumsi bandwidth dengan segmentasi video dan penghapusan otomatis segmen lama. Selain mendukung inisiatif *smart city* melalui layanan pemantauan lalu lintas *real-time* yang terbuka, arsitektur sistem ini juga dapat direplikasi oleh daerah lain secara efisien dengan pendekatan *open-source*, sekaligus menjadi dasar evaluasi teknis untuk pengembangan lanjutan di masa mendatang.

2. METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimental terapan yang bertujuan untuk mengevaluasi implementasi teknologi HTTP Live Streaming (HLS) pada sistem pemantauan CCTV publik di Kota Pekanbaru. Pendekatan ini dipilih untuk memungkinkan pengujian langsung terhadap arsitektur sistem streaming yang dikembangkan, sekaligus memperoleh data empiris dari proses operasional yang berlangsung secara *real-time* di lingkungan nyata.

Langkah-langkah penelitian dimulai dari identifikasi permasalahan pada sistem pemantauan eksisting berbasis RTSP, kemudian dilanjutkan dengan perancangan dan penerapan sistem HLS menggunakan komponen-komponen open-source seperti *FFmpeg*, *Nginx*, *Supervisord*, dan HLS.js. Setelah implementasi selesai, dilakukan proses monitoring performa

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

sistem melalui serangkaian pengukuran teknis seperti konsumsi *bandwidth*, kompatibilitas lintas perangkat, dan stabilitas proses *streaming*.

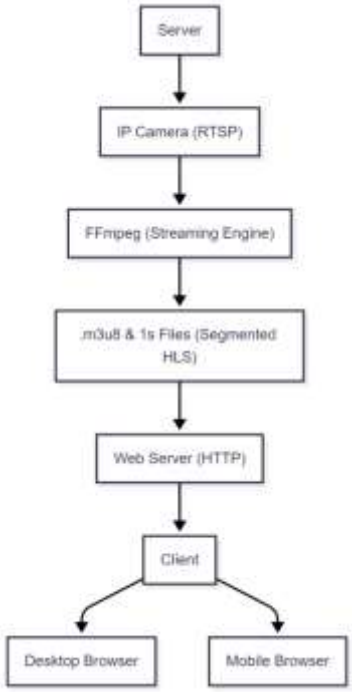
Penelitian ini juga mengevaluasi kemampuan sistem dalam menangani multi-kamera secara paralel, serta menganalisis sejauh mana sistem dapat mendukung kebutuhan pemantauan publik yang stabil, efisien, dan mudah diakses oleh masyarakat melalui berbagai platform. Untuk mendukung validitas data, digunakan berbagai alat bantu seperti traffic monitoring (*bmon*), serta observasi pemutaran streaming di berbagai browser modern.

Metode penelitian ini dijelaskan secara terstruktur dalam beberapa sub bagian, dimulai dari arsitektur sistem, desain alur HLS, perangkat lunak yang digunakan, metode pengukuran performa, hingga pengujian kompatibilitas klien.

1. Diagram Arsitektur Sistem Streaming HLS

Sistem streaming yang dikembangkan oleh Pemerintah Kota Pekanbaru untuk layanan pemantauan CCTV publik menggunakan pendekatan berbasis HTTP Live Streaming (HLS). Arsitektur sistem ini dirancang agar mampu menerima aliran video dari berbagai kamera pengawas yang tersebar di titik-titik strategis kota, melakukan konversi menjadi segmen video HLS, dan menyajikannya kepada publik secara real-time melalui web browser.

Secara umum, arsitektur sistem streaming HLS terdiri dari beberapa komponen utama, seperti terlihat pada gambar berikut:



Gambar 1. Arsitektur sistem streaming HLS

penjelasan komponen yang ada pada arsitektur:

a. Kamera CCTV (RTSP Stream Source)
Setiap kamera yang terpasang di lapangan menghasilkan output video menggunakan protokol RTSP (*Real-Time Streaming Protocol*). RTSP (*Real-Time Streaming Protocol*) merupakan protokol aplikasi yang umum digunakan dalam pertukaran data streaming multimedia. Protokol ini berfungsi sebagai pengendali komunikasi *real-time* antara perangkat pengirim, seperti kamera IP, dengan perangkat pemantau, memungkinkan transmisi video secara langsung(Rizki et al., 2019).

b. FFmpeg (*Transcoder*):
Perangkat lunak FFmpeg digunakan sebagai mesin pengolah (transcoder) yang mengonversi aliran video RTSP menjadi format HLS. Dalam proses ini, FFmpeg melakukan segmentasi video menjadi file berformat .ts dan menyusun file indeks .m3u8 sebagai playlist utama. Peran FFmpeg dalam arsitektur sistem siaran langsung berbasis perangkat lunak terbuka telah diteliti dan dibuktikan dalam konteks sistem streaming kampus(Liu et al., 2023).

c. Web Server (Nginx):
File hasil segmentasi HLS disimpan di direktori khusus yang dapat diakses publik melalui web server HTTP. Web server ini (misalnya Nginx atau Apache) tidak perlu komponen streaming khusus, karena HLS

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

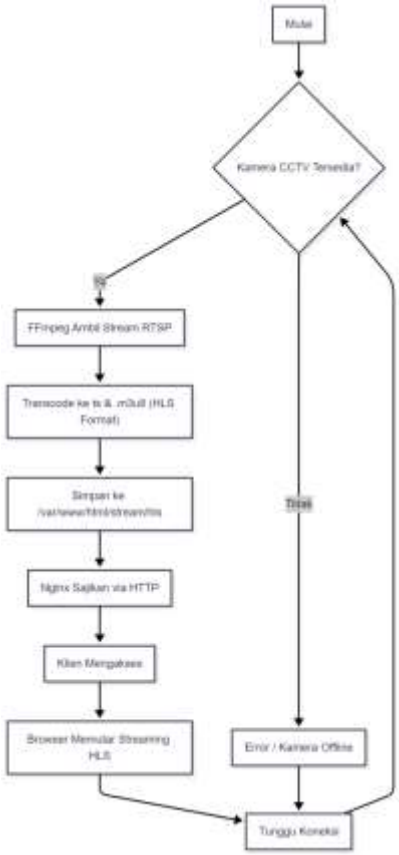
hanya memerlukan protokol HTTP standar. Nginx berperan sebagai web server HTTP yang menangani permintaan klien dan kinerjanya dapat ditingkatkan melalui berbagai pengoptimalan, seperti cache dan pengaturan proses, sehingga menghasilkan peningkatan performa signifikan dalam alur pengolahan HTTP (Wang & Kai, 2021).

d. Client (User Browser):
Klien dapat mengakses video melalui browser baik di perangkat *desktop* maupun *mobile*. Pemutar video modern di browser seperti Chrome, Firefox, Edge, dan Safari dapat menggunakan pustaka HLS berbasis JavaScript seperti hls.js atau hasplayer.js yang memanfaatkan Media Source Extensions (MSE) untuk melakukan transmuxing aliran HLS langsung di sisi klien. Dengan pendekatan ini, pemutaran video dapat dilakukan melalui HTTP standar tanpa memerlukan plugin eksternal (Silhavy et al., 2017).

e. Karakteristik Sistem:
   1) Arsitektur terdistribusi, dengan banyak proses *FFmpeg* yang berjalan secara paralel, masing-masing menangani satu kamera.
   2) Skalabilitas horizontal, memungkinkan penambahan kamera baru tanpa mengubah arsitektur inti.
   3) Berbasis *open-source*, dengan penggunaan *FFmpeg*, *supervisord*, dan *Nginx* untuk menekan biaya pengembangan.

2. Desain Sistem HLS (Diagram Alur)

Proses alur kerja sistem streaming HLS pada layanan CCTV Kota Pekanbaru dibangun secara modular, mulai dari akuisisi sumber video hingga penyajian stream ke sisi klien. Untuk memastikan keteraturan, keandalan, dan pemantauan yang mudah, setiap proses dikontrol melalui konfigurasi sistem dan dijalankan otomatis oleh aplikasi *supervisor*. Diagram alur berikut menggambarkan tahapan proses secara berurutan:



Gambar 2. Tahapan proses streaming

Berikut penjelasan alur proses

1. Pengecekan Kamera Aktif:
   Proses diawali dengan mendeteksi apakah kamera CCTV pada endpoint tertentu tersedia dan dapat diakses melalui alamat RTSP.
2. Transkoding oleh FFmpeg:
   Jika koneksi RTSP berhasil, aplikasi FFmpeg akan mengambil *stream* dan melakukan konversi (*transcoding*) ke format HLS, yakni file .ts (video segment) dan .m3u8 (playlist).
3. Penyimpanan File HLS:
   File hasil *transcoding* disimpan ke direktori sementara (*/var/www/html/stream/hls/* atau lokasi serupa) yang telah dikonfigurasi agar dapat diakses oleh publik.
4. Distribusi via Web Server:
   Server web (misalnya Nginx) berperan menyajikan file .m3u8 dan .ts kepada pengguna. Karena HLS hanya menggunakan HTTP standar, tidak diperlukan teknologi streaming khusus.
5. Pemutaran oleh Klien:
   Saat pengguna membuka halaman CCTV melalui browser (baik *desktop* maupun

I N F O R M A T I K A
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

mobile), player berbasis HLS.js akan secara otomatis membaca file .m3u8, mengunduh segmentasi .ts, dan menampilkan video secara real-time.

Keunggulan Desain:

1. Tangguh dan modular, karena tiap kamera dikelola oleh proses FFmpeg terpisah yang dijalankan oleh *supervisord*
2. Efisien, karena hanya segmen video terakhir yang disimpan, dan segmen lama dihapus otomatis
3. Fleksibel, karena format HTTP kompatibel dengan semua *browser* tanpa plugin tambahan

3. Tools FFmpeg, Nginx, Supervisord

Dalam pengembangan dan pengujian sistem streaming CCTV berbasis HLS pada Kota Pekanbaru, digunakan beberapa perangkat lunak utama yang bersifat *open-source*, andal, dan kompatibel dengan sistem operasi Linux. Setiap tools memiliki peran spesifik mulai dari pengambilan video, konversi format, hingga mempertahankan *transcoder* selalu aktif.

1. FFmpeg
   FFmpeg merupakan aplikasi berupa command *line* yang digunakan untuk memproses video dan audio, termasuk konversi antar format, *streaming real-time*, serta pengolahan *codec* (Zeng & Fang, 2013). Dalam sistem ini, *FFmpeg* bertanggung jawab untuk:
   a. Mengambil input dari kamera CCTV menggunakan protokol RTSP
   b. Melakukan transkoding video menjadi format HLS, yaitu .m3u8 dan .ts
   c. Menentukan parameter segmentasi seperti -hls_time, -hls_list_size, dan -hls_flags
   d. Menyimpan output pada direktori yang bisa diakses oleh Nginx

   Berikut adalah perintah *ffmpeg* yang digunakan untuk transkoding dari input rtsp menjadi segmentasi hls:

```
ffmpeg -loglevel warning -fflags
nobuffer -threads 4 -rtsp_transport
tcp -i
rtsp://guest:password@10.15.4.2:554/
stream1  -c:v libx264 -preset
ultrafast -tune zerolatency -
profile:v baseline -level 3.1 -x264-
params "keyint=30:min-keyint=30:no-
scenecut:intra-refresh=1:nal-
hrd=cbr" -pix_fmt yuv420p -g 30 -r
30 -b:v 1500k -maxrate 1500k -
bufsize 1000k -threads 2 -s 640x360
-b:v 1000k -maxrate 1000k -bufsize
500k -an -f flv -flvflags
no_duration_filesize+no_sequence_end
rtmp://localhost/live/cctv-1204
```

Nginx digunakan sebagai server HTTP untuk menyajikan file streaming hasil transkoding HLS kepada pengguna. Keunggulan Nginx antara lain:

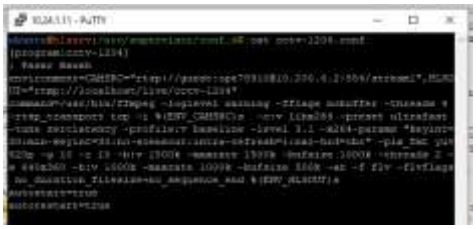
a) Ringan dan efisien dalam menangani trafik HTTP berkapasitas tinggi
b) Tidak memerlukan modul tambahan untuk menyajikan file .m3u8 atau .ts
c) Mudah dikonfigurasi untuk integrasi dengan direktori output FFmpeg

Dalam arsitektur ini, direktori html seperti */var/www/html/stream/hls/* disiapkan sebagai root akses publik Nginx, sehingga file streaming dapat dimuat oleh player berbasis browser.

2. Supervisord
   Supervisord adalah sistem manajemen proses yang memungkinkan administrator menjalankan, memantau, dan mengelola banyak program secara bersamaan secara otomatis. Dalam konteks implementasi HLS untuk CCTV Kota Pekanbaru, *supervisord* digunakan untuk:
   a. Menjalankan banyak proses FFmpeg secara paralel, masing-masing untuk satu kamera RTSP
   b. Memastikan bahwa proses tetap berjalan secara terus-menerus (jika FFmpeg crash, akan otomatis direstart)
   c. Menyediakan log terstruktur per proses untuk debugging
   d. Mengorganisasi konfigurasi dalam file .conf per kamera (misalnya cctv-1201.conf, cctv-1202.conf, dst)

   Berikut adalah contoh isi file konfigurasi supervisord:

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694



Gambar 3. Konfigurasi supervisord

Dengan pendekatan ini, sistem menjadi lebih modular, *fault-tolerant*, dan mudah diperluas hanya dengan menambahkan file konfigurasi baru tanpa memodifikasi arsitektur utama.

3. HLS.js

HLS.js adalah pustaka JavaScript berbasis HTML5 yang digunakan untuk memutar streaming HLS di browser yang tidak mendukungnya secara native (misalnya Chrome, Firefox, Edget, Opera). Meskipun browser seperti Safari mendukung .m3u8 secara default, sebagian besar browser modern memerlukan bantuan pustaka ini agar dapat membaca dan memutar stream .ts secara real-time.

Fungsi HLS.js dalam sistem:

1) Memuat file .m3u8 dari web server
2) Mengelola buffer stream video secara otomatis
3) Menangani error koneksi dan auto-reconnect
4) Mendukung adaptive bitrate streaming (jika diaktifkan)

Berikut adalah contoh kode untuk integrasi HLS.js dalam HTML:

```
    <video id="video" controls
autoplay></video>
    <script
src="https://cdn.jsdelivr.net/npm/hls.js@lates
t"></script>
    <script>
      if (Hls.isSupported()) {
        var video =
document.getElementById('video');
        var hls = new Hls();

hls.loadSource('cctv.pekanbaru.go.id/stream/cc
tv-1201.m3u8');
        hls.attachMedia(video);
        hls.on(Hls.Events.MANIFEST_PARSED,
function() {
          video.play();
        });
      }
    </script>
```

Dengan menggunakan HLS.js, **pengguna tidak memerlukan ekstensi tambahan** atau konfigurasi khusus untuk menonton stream cukup membuka situs dan memilih kamera yang diinginkan.

Integrasi Antar Tools

1. *Supervisord* bertugas menjaga proses FFmpeg tetap aktif dan berjalan stabil
2. *FFmpeg* menangani pengolahan video menjadi format HLS
3. *Nginx* menyajikan file hasil HLS
4. *HLS.js* menjalankan pemutar video di sisi klien (browser)

4. Pengukuran (*Latency*, Bandwidth, dan Kompatibilitas Klien)

Untuk menilai efektivitas sistem streaming CCTV berbasis HLS yang diterapkan oleh Pemerintah Kota Pekanbaru, dilakukan pengukuran terhadap beberapa metrik yang mencerminkan kinerja sistem dari sisi teknis maupun pengalaman pengguna. Pengukuran dilakukan secara deskriptif menggunakan data *real-time* serta observasi langsung dari sisi server dan klien.

Metrik utama yang digunakan dalam evaluasi ini adalah: *latency* (waktu tunda video), *bandwidth* (penggunaan data jaringan), dan kompatibilitas di sisi klien.

a. Latency

Latency didefinisikan sebagai selisih waktu antara kejadian di lapangan (misalnya kendaraan lewat kamera) dan saat kejadian tersebut ditampilkan di layar pengguna.

Pengukuran dilakukan dengan cara observasi visual terhadap:

1. Tampilan langsung di lokasi kamera (real view)
2. Tampilan video streaming dari browser

Gambar 4. Waktu tunda tampilan hls vs rtsp

Dari gambar sample yang diambil, pada sisi kiri terlihat timestamp dari streaming rtsp (10:11:09) sedangkan timestamp streaming hls 10:11:05), terlihat ada perbedaan sebanyak 4 detik, hls mengalami sekitar 4 detik penundaan.

Berikut adalah data *timestamp* yang diambil dari tampilan pada tangkapan layar video seperti diatas pada *slot* waktu yang berbeda-beda:

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

Tabel 1. Hasil observasi *visual latency*

No.	RTSP	HLS	Selisih (detik)
1	04:48:08	04:48:03	+5
2	06:06:48	06:06:43	+5
3	06:32:25	06:32:21	+4
4	06:34:58	06:34:53	+5
5	06:36:37	06:36:33	+4
6	06:37:32	06:37:28	+4
7	06:43:34	06:43:29	+5
8	06:44:18	06:44:14	+4
9	18:17:04	18:17:04	0
10	18:57:00	18:56:55	+5
11	18:57:29	18:57:29	0
12	18:57:53	18:57:51	+2
13	19:15:55	19:15:55	0
		Average	3.307
		Median	4

Dari sampel data yang diambil secara real, terlihat latency yang terjadi pada hls adalah sekitar 4 detik. Waktu tunda diamati rata-rata dalam kisaran 2–5 detik menggunakan HLS (dengan menggunakan nilai default hls_time=2s)

Namun, meskipun HLS memiliki *latency* lebih tinggi, keuntungan dari sisi aksesibilitas browser dan kestabilan membuatnya lebih unggul dalam skenario publik.

b. Bandwidth

Penggunaan bandwidth diamati melalui monitoring menggunakan aplikasi *bmon* yang tersedia pada Linux:

Gambar 5. Monitoring bandwidth (bmon)

Observasi menunjukka streaming satu kamera dengan HLS memerlukan sekitar 2.5 Mbps secara konstan

Segmentasi ini juga memungkinkan pengurangan *buffer overrun*, serta mencegah pengguna mengunduh stream berlebihan saat koneksi tidak stabil. Segmentasi media dalam MPEG-DASH memungkinkan klien memilih segmen video berdasarkan kondisi jaringan secara *real-time*, yang mencegah kelebihan unduhan saat bandwidth rendah dan membantu menjaga pengalaman pemutaran yang lancar. Hal ini dilakukan dengan menyesuaikan kualitas segmen di setiap batas segmen untuk menghindari buffer overrun dan menghemat bandwidth (Mueller et al., 2013).

c. Kompatibilitas (Browser/Desktop/Mobile)

Pengujian dilakukan terhadap sejumlah browser populer di berbagai platform:

Tabel 2. Hasil pengujian HLS.js

Platform	Chrome	Firefox	Edge	Safari	Opera
Windows	✔	✔	✔	✘	✔
Android	✔	✔	✔	✘	✔
macOS	✔	✔	✔	✔	✔
iOS (Safari)	✔	✘	✘	✔	✘

Keterangan:

1) ✔ : Kompatibel, berhasil memutar stream .m3u8
2) ✘ : Tidak kompatibel langsung tanpa pustaka
3) Untuk browser lain, pustaka HLS.js digunakan agar pemutaran berhasil

Secara keseluruhan, kompatibilitas tergolong tinggi, dengan syarat halaman web mengintegrasikan player berbasis JavaScript modern seperti HLS.js.

d. Stabilitas dan Uptime

Metrik tambahan yang diamati:

1) Stabilitas proses FFmpeg (melalui supervisord)
2) Recovery otomatis dalam 5 detik bila koneksi kamera terputus

Pengukuran terhadap latency, bandwidth, dan kompatibilitas klien menunjukkan bahwa sistem streaming HLS yang diterapkan sudah **cukup efisien dan handal** untuk kebutuhan pemantauan kota secara publik. Meskipun latency tidak serendah RTSP, namun kestabilan dan kemudahan akses menjadikan HLS sebagai solusi ideal untuk sistem streaming skala kota.

e. Pengujian Kompatibilitas di Client Browser (*Desktop* dan Mobile)

Salah satu keunggulan utama penggunaan protokol HTTP Live Streaming (HLS) dalam sistem pemantauan CCTV adalah kemudahan akses lintas perangkat dan lintas *platform*, tanpa memerlukan aplikasi tambahan atau plugin khusus. Untuk memastikan hal ini, dilakukan serangkaian pengujian kompatibilitas streaming terhadap berbagai kombinasi browser dan sistem operasi.

Pengujian dilakukan dengan skenario pengguna membuka alamat streaming CCTV publik melalui antarmuka web (cctv.pekanbaru.go.id) menggunakan perangkat desktop maupun mobile, dengan dan tanpa dukungan pustaka HLS.js.

a. Parameter Pengujian

1) Browser yang diuji: Chrome, Firefox, Edge, Safari, Opera
2) Perangkat yang digunakan (Laptop Windows, HP Android, iPhone iOS)
3) Metode pemutaran: Pemutar HTML5 dengan dan tanpa integrasi HLS.js
4) Jenis koneksi: Wi-Fi dan jaringan seluler

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694

b. Hasil Pengujian

Tabel berikut menunjukkan hasil uji kompatibilitas pemutaran stream .m3u8 di berbagai kombinasi:

Tabel 3. Hasil uji lebih lengkap

Perangkat	OS	Browser	Native	HLS.js	Status Pemutaran
Laptop	Win	Chrome	✗	✓	Berhasil
Laptop	Win	Firefox	✗	✓	Berhasil
Laptop	Win	Edge	✗	✓	Berhasil
Laptop	Win	Opera	✗	✓	Berhasil
HP	Android	Chrome	✗	✓	Berhasil
HP	Android	Firefox	✗	✓	Berhasil
HP	Android	Opera	✗	✓	Berhasil
iPhone	iOS	Safari	✓ sebagian	✓	Berhasil
iPhone	iOS	Chrome/ Edge	✗	✓	Berhasil
macBook	macOS	Safari	✓ sebagian	✓	Berhasil
macBook	macOS	Chrome/ Edge	✗	✓	Berhasil

1) Browser modern seperti Chrome, Firefox, dan Edge tidak mendukung pemutaran HLS secara native, tetapi dapat memutar video .m3u8 dengan sangat baik melalui pustaka HLS.js.
2) Sebagian Safari di iOS dan macOS mendukung HLS secara bawaan (native) tanpa memerlukan pustaka tambahan, sehingga pengalaman pengguna lebih ringan.
3) Browser iOS non-Safari (misalnya Chrome, Firefox di iOS) tidak dapat memutar stream HLS, karena dibatasi oleh WebKit engine milik Apple.

Dalam hal kompatibilitas ini, untuk memastikan hasil yang maksimal, sistem frontend web streaming cctv.pekanbaru.go.id telah menerapkan deteksi otomatis terhadap ketersediaan dukungan HLS:

```
    if (!Hls.isSupported()) {
        video.src = source;
    } else {
        const hls = new Hls();
        hls.loadSource(source);
        hls.attachMedia(video);
        window.hls = hls;

        player.on('ready', event => {
            const instance = event.detail.plyr;
            var el = this.frame.find(".plyr__controls").children().first();
            el.after("<div class=''>" + "<span class='d-none'> this.cam.vlan </span>" + " <div class='badge badge-danger badge-outline'> LIVE NOW </div> " + "</div>");
        });
        player.on('error', event => {
            console.log('Plyr Error');
        });
    }
```

Dengan logika tersebut, pengguna dari hampir semua platform dapat menikmati stream CCTV tanpa hambatan teknis.

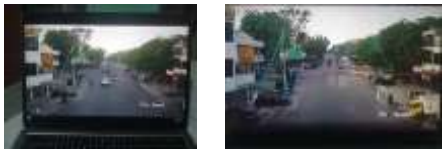
Hasil pengujian menunjukkan bahwa sistem HLS yang dikembangkan telah berhasil memberikan pengalaman pemantauan video real-time yang kompatibel, stabil, dan efisien di berbagai perangkat. Integrasi pustaka HLS.js sangat krusial untuk memastikan pemutaran berjalan baik di browser modern non-Safari.

3. HASIL DAN PEMBAHASAN

Hasil dari implementasi HTTP Live *Streaming* (HLS) pada sistem pemantauan CCTV Kota Pekanbaru menunjukkan bahwa pendekatan ini tidak hanya kompatibel dan mudah diakses oleh masyarakat, tetapi juga menunjukkan performa teknis yang stabil di sisi server. Pembahasan dalam bab ini difokuskan pada tiga aspek utama: hasil implementasi sistem, analisis performa teknis, dan komparasi RTSP vs. HLS.

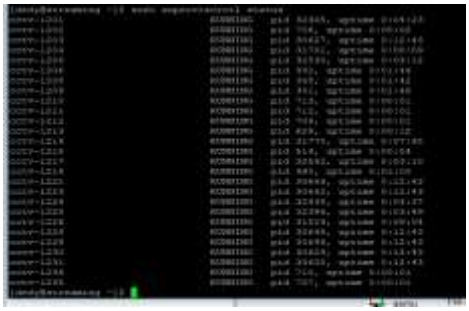
a. Implementasi Sistem

Sistem pemantauan CCTV dapat diakses melalui laman publik cctv.pekanbaru.go.id, di mana sejumlah kamera dari berbagai lokasi penting di Kota Pekanbaru disajikan secara *real-time*. Setiap kamera ditangani oleh satu proses FFmpeg yang dijalankan dan diawasi oleh *supervisord*, menghasilkan stream .m3u8 dan .ts yang disimpan di direktori sementara (/var/www/html/stream/hls). Beberapa konfigurasi sudah di-optimalisasi lebih lanjut, seperti mengurangi frame size sehingga menjadi lebih hemat *bandwidth*. Berikut adalah beberapa tangkapan layar sistem:



Gambar 6. Tampilan macBook dan Windows

INFORMATIKA
Jurnal Informatika, Manajemen dan Komputer, Vol. 18 No. 1, Mei 2026
eISSN : 2580-3042
pISSN : 1979-0694



Gambar 7. *supervisord process list*

b. Perbandingan RTSP vs HLS

Perbandingan antara teknologi RTSP dan HLS pada aspek-aspek utama ditampilkan dalam tabel berikut:

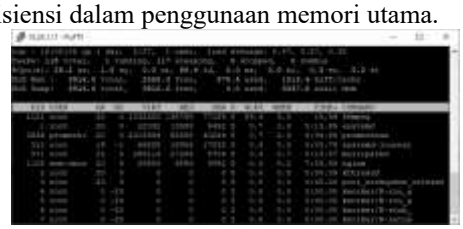
Tabel 4. Perbandingan RTSP vs HLS

Aspek	RTSP	HLS
Latency	Rendah (<1 detik)	Sedang (3–7 detik)
Kompatibilitas	Terbatas (butuh VLC/plugin)	Luas (via browser + HLS.js)
Keandalan Server	Tidak teroptimasi	Stabil dengan auto-restart (mis. *supervisord*)
Akses Publik	Sulit jika tanpa konfigurasi	Mudah melalui HTTP standard

HLS terbukti lebih unggul dalam skenario publik dan multi-platform meskipun memiliki latency lebih tinggi dibanding RTSP.

c. Penggunaan CPU

Berdasarkan hasil observasi sistem melalui perintah lscpu dan top, diketahui bahwa server HLS yang digunakan berjalan di atas platform virtualisasi KVM dengan arsitektur x86_64, menggunakan dua inti logis prosesor dari keluarga Intel. Sistem ini dilengkapi dengan 4 GB RAM dan tidak memanfaatkan ruang swap aktif saat pengujian dilakukan, menunjukkan efisiensi dalam penggunaan memori utama.



Gambar 8. Server *load* menjalankan ffmpeg

Proses transkoding dilakukan oleh perangkat lunak FFmpeg, yang secara aktif mengonversi aliran video dari kamera IP berbasis RTSP menjadi potongan segmen .ts sesuai spesifikasi protokol HTTP Live Streaming (HLS). FFmpeg juga membangkitkan file indeks .m3u8 untuk pengendalian urutan pemutaran segmen. Berdasarkan data pemantauan proses, FFmpeg menunjukkan penggunaan CPU yang cukup signifikan, yaitu sekitar 57.7% dari total kapasitas CPU, menandakan bahwa proses encoding ini sangat intensif secara komputasional.

Meski demikian, sistem tetap menunjukkan performa yang stabil dengan nilai load average yang masih dalam kisaran aman, yakni 1.84 (1 menit), 1.15 (5 menit), dan 0.99 (15 menit). Hal ini menunjukkan bahwa sistem mampu mempertahankan beban kerja berkelanjutan tanpa mengalami bottleneck atau backlog proses. Selain itu, penggunaan memori juga terjaga dengan baik, di mana hanya sekitar 1.3 GB dari total 3.9 GB RAM yang digunakan secara aktif.

Secara keseluruhan, pengujian ini menunjukkan bahwa implementasi FFmpeg dalam proses transkoding HLS pada server berspesifikasi ringan ini mampu berjalan dengan stabil dan efisien, meskipun terbatas pada dua inti CPU. Hal ini mengindikasikan bahwa arsitektur sistem saat ini cukup optimal untuk mendukung pemantauan video *multi-stream* berbasis web melalui protokol HLS.

d. Dampak Operasional

Dengan sistem ini, dapat diambil beberapa makna penting yang dilakukan oleh Pemerintah Kota Pekanbaru:

1) Memberikan akses informasi lalu lintas secara transparan kepada publik
2) Mendukung agenda *smart city* dalam aspek keterbukaan dan keamanan
3) Menerapkan solusi yang efisien, scalable, dan berbasis *open-source*

4. KESIMPULAN

Penelitian ini telah mengevaluasi penerapan protokol HTTP Live Streaming (HLS) pada sistem pemantauan CCTV publik Kota Pekanbaru, yang tersedia secara daring melalui alamat cctv.pekanbaru.go.id. Dari hasil observasi, analisis teknis, dan pengujian lintas platform, dapat disimpulkan bahwa sistem streaming berbasis HLS berhasil dibangun menggunakan kombinasi perangkat lunak open-source seperti FFmpeg, Nginx, dan supervisord. Pendekatan ini terbukti mampu menangani lebih kamera RTSP dengan stabilitas yang baik. Melalui integrasi pustaka HLS.js, video streaming dapat diakses langsung dari hampir semua browser modern, baik di perangkat desktop maupun mobile, tanpa perlu aplikasi tambahan. Hal ini memperkuat akses informasi yang inklusif bagi masyarakat. Dari sisi performa teknis, sistem menunjukkan kestabilan

jaringan dengan konsumsi rata-rata berkisar 2 hingga 3 Mbps per kamera. Segmentasi video yang pendek, yakni berdurasi dua detik, memberikan pengalaman menonton yang nyaris seketika. Jika dibandingkan dengan RTSP, protokol HLS menawarkan keunggulan dari segi aksesibilitas dan keandalan sistem, meskipun memiliki *latency* yang sedikit lebih tinggi. Keunggulan ini menjadikan HLS lebih sesuai untuk kebutuhan publik berskala besar. Secara keseluruhan, implementasi teknologi ini mendukung konsep *smart city*, dengan menekankan pada keterbukaan data, layanan berbasis teknologi informasi, serta efisiensi pemanfaatan infrastruktur digital oleh pemerintah daerah.

Rekomendasi

1. Pemerintah daerah lain dapat mereplikasi pendekatan ini untuk meningkatkan transparansi dan keterbukaan informasi publik melalui layanan CCTV berbasis HLS.
2. Untuk pengembangan selanjutnya, sistem dapat diperluas dengan fitur tambahan seperti:
   a) Deteksi objek berbasis AI
   b) Statistik pengunjung atau kendaraan
   c) Integrasi dengan dashboard keamanan terpadu
3. Perlu dilakukan optimasi performa sistem, misalnya dengan menggunakan GPU untuk transkoding atau membagi beban ke beberapa server (*horizontal scaling*).

5. REFERENSI

Alkaf, A. M., & Sutrisno, B. (2019). Smart Surveillance Dan Keteraturan Sosial (Studi Kasus Implementasi Smart City Di Kota Bandung). *Jurnal Sosioteknologi*, *18*(1), 91–105. https://doi.org/10.5614/sostek.itbj.2019.18.1.7

Almegi, A., & Fatmawati, F. (2024). Perkembangan Lahan Terbangun Kota Pekanbaru Tahun 2000-2020. *Geomedia Majalah Ilmiah Dan Informasi Kegeografian*, *21*(2), 190–201. https://doi.org/10.21831/gm.v21i2.65372

KUMAR, D. (2024). Actual practices of citizen participation in smart cities. *Smart Cities and Regional Development (SCRD) Journal*, *8*(2), 19–30. https://doi.org/10.25019/4c05yr24

Laufs, J., Borrion, H., & Bradford, B. (2020). Security and the smart city: A systematic review. *Sustainable Cities and Society*, *55*. https://doi.org/10.1016/j.scs.2020.102023

Liu, C., Mao, Q., Wang, Y., Liu, J., & Wang, Y. (2023). Analysis and Research of Campus Live Broadcasting System Based on SRS and FFmpeg. *ACM International Conference Proceeding Series*, 1306–1310. https://doi.org/10.1145/3650400.3650620

Mueller, C., Lederer, S., Poecher, J., & Timmerer, C. (2013). Demo paper: Libdash - An open source software library for the MPEG-DASH standard. *Electronic Proceedings of the 2013 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2013*, 1–2. https://doi.org/10.1109/ICMEW.2013.6618220

Nunez, L., & Toasa, R. M. (2020). Performance evaluation of RTMP, RTSP and HLS protocols for IPTV in mobile networks. *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, *2020-June*, 1–5. https://doi.org/10.23919/CISTI49556.2020.9140848

Rizki, R., Munadi, R., & Syahrial, S. (2019). Analisis Performansi Video Streaming Dengan Menggunakan Protokol RTSP Pada Jaringan IEEE 802.11n. *Jurnal Nasional Komputasi Dan Teknologi Informasi (JNKTI)*, *2*(1), 9. https://doi.org/10.32672/jnkti.v2i1.1050

Saini, S. S., & Sharma, L. Sen. (2024). Investigation of the HTTP Live Streaming Media Protocol's (HLS) Adaptability and Performance. *Journal of The Institution of Engineers (India): Series B*. https://doi.org/10.1007/s40031-024-01132-w

Seufert, M., Egger, S., Slanina, M., Zinner, T., Hoßfeld, T., & Tran-Gia, P. (2015). A Survey on Quality of Experience of HTTP Adaptive Streaming. *IEEE Communications Surveys and Tutorials*, *17*(1), 469–492. https://doi.org/10.1109/COMST.2014.2360940

Silhavy, D., Pham, S., & Arbanowski, S. (2017). Performance considerations of HTML5-based dynamic packaging for media Streaming. *Proceedings of the 27th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV 2017*, 7–12. https://doi.org/10.1145/3083165.3083172

Stockhammer, T. (2011). Dynamic adaptive streaming over HTTP --. *Proceedings of*

the Second Annual ACM Conference on Multimedia Systems, *January*, 133–144. https://doi.org/10.1145/1943552.1943572

Thanh Le, T., Jeong, J., & Ryu, E.-S. (2019). Efficient Transcoding and Encryption for Live 360 CCTV System. *Applied Sciences*, *9*(4), 760. https://doi.org/10.3390/app9040760

Wang, J., & Kai, Z. (2021). Performance Analysis and Optimization of Nginx-based Web Server. *Journal of Physics: Conference Series*, *1955*(1), 0–6. https://doi.org/10.1088/1742-6596/1955/1/012033

Yang, S., Nakajima, H., Yang, Y., Shin, Y., & Koizumi, H. (2024). The impact of surveillance cameras and community safety activities on crime prevention: Evidence from Kakogawa City, Japan. *Sustainable Cities and Society*, *115*. https://doi.org/10.1016/j.scs.2024.105858

Zeng, H., & Fang, Y. (2013). Implementation of video transcoding client based on Ffmpeg. *Advanced Materials Research*, *756–759*, 1748–1752. https://doi.org/10.4028/www.scientific.net/AMR.756-759.1748

Zhang, J., Wang, Z. J., Guo, S., Yang, D., Fang, G., Peng, C., & Guo, M. (2018). Power consumption analysis of video streaming in 4G LTE networks. *Wireless Networks*, *24*(8), 3083–3098. https://doi.org/10.1007/s11276-017-1519-9