SISTEM CCTV BERBASIS IOT LIVE STREAM DENGAN PENGAMBILAN GAMBAR PADA PT. MEKA PRIMA SOLUSI

KERJA PRAKTIK

Program Studi

S1 Teknik Komputer

Oleh :

Mark Felix

22410200003

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2025

SISTEM CCTV BERBASIS IOT LIVE STREAM DENGAN PENGAMBILAN GAMBAR PADA PT. MEKA PRIMA SOLUSI

Diajukan sebagian salah satu syarat untuk menyelesaikan

Program Strata Satu (S1)

Disusun Oleh :

Nama : Mark Felix

NIM : 22410200003

Program : S1 (Strata Satu)

Jurusan : Teknik Komputer

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2025

“Always Do Your Best.”

LEMBAR PENGESAHAN

LEMBAR PENGESAHAN

SISTEM CCTV BERBASIS IOT LIVE STREAM DENGAN PENGAMBILAN GAMBAR PADA PT.MEKA PRIMA SOLUSI

Laporan Kerja Praktik oleh

Mark Felix

NIM : 22410200003

Telah diperiksa, diuji dan disetujui

Surabaya, 30 Agustus 2025

Disetujui:

Pembimbing	Penyelia
Dr. Ira Puspasari, S.Si., M.T.,	**Christian**
NIDN. 0710078601	

Mengetahui,
Ketua Program Studi S1 Teknik Komputer

Fakultas Teknologi dan Informatika
UNIVERSITAS Dinamika

Dr. Ira Puspasari, S.Si., M.T.
NIDN. 0710078601

LEMBAR PERNYATAAN

PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa **Universitas Dinamika**, Saya :

Nama	: **Mark Felix**
NIM	: **22410200003**
Program Studi	: **S1 Sistem Komputer**
Fakultas	: **Fakultas Teknologi dan Informatika**
Jenis Karya	: **Laporan Kerja Praktik**
Judul Karya	: **SISTEM CCTV BERBASIS IOT LIVE STREAM DENGAN PENGAMBILAN GAMBAR PADA PT. MEKA PRIMA**

Menyatakan dengan sesungguhnya bahwa :

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, Saya menyetujui memberikan kepada **Universitas Dinamika** Hak Bebas Royalti Non-Eksklusif (***Non-Exclusive Royalty Free Right***) atas seluruh isi/sebagian karya ilmiah Saya tersebut diatas untuk disimpan, dialihmediakan, dan dikelola dalam bentuk pangkalan data (***database***) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama Saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut diatas adalah hasil karya asli Saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya, atau pendapat orang lain yang ada dalam karya ilmiah ini semata-mata hanya sebagai rujukan yang dicantumkan dalam Daftar Pustaka Saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiasi pada karya ilmiah ini, maka Saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada Saya.

Demikan surat pernyataan ini Saya buat dengan sebenar-benarnya.

Surabaya, 30 Agustus 2025

Mark Felix
NIM : 22410200003

ABSTRAK

Penelitian ini berfokus pada pengembangan dan penerapan CCTV *live stream* yang terintegrasi dengan *IoT* di lingkungan PT. Meka Prima Solusi, dengan tujuan meningkatkan efektivitas pengawasan. Sistem ini memanfaatkan ESP32-CAM, sebuah *microcontroller* yang hemat biaya namun sudah dilengkapi kapabilitas *Wi-Fi* dan kamera terintegrasi. Perangkat ringan ini berfungsi sebagai unit akuisisi dan pemroses gambar utama, menangkap tayangan video langsung dan gambar diam, lalu mentransmisikannya melalui jaringan nirkabel. Fitur pengambilan gambar diam dapat diotomatisasi berdasarkan waktu atau diaktifkan langsung oleh pengguna, dan pemantauan dapat diakses melalui antarmuka web. Implementasi sistem ini menunjukkan keberhasilan dalam menyajikan pengawasan visual *real-time* yang sekaligus hemat energi dan biaya. Solusi *IoT* ini secara nyata meningkatkan efisiensi operasional dan kesiapsiagaan keamanan.

Kata kunci: cctv, iot, snapshot

KATA PENGANTAR

Laporan riset berjudul "Penerapan Sistem CCTV *Live Stream* Berbasis *IoT* dengan Fitur Pengambilan Gambar di PT. Meka Prima Solusi" ini akhirnya rampung berkat rahmat dan anugerah dari Tuhan Yang Maha Esa. Penyusunan dokumen ini merupakan salah satu syarat wajib untuk menyelesaikan tugas laporan Kerja Praktik (KP) di program studi Teknik Komputer.

Seiring pesatnya kemajuan teknologi *Internet of Things* (IoT), muncul peluang untuk menciptakan sistem keamanan yang lebih pintar dan berdaya guna tinggi. Proyek ini secara khusus berfokus pada perancangan dan implementasi sistem kamera pengawas (CCTV) menggunakan modul ESP32-CAM. Sistem ini dirancang untuk dapat menyajikan tayangan video secara langsung (*live stream*) sekaligus mengambil gambar statis, semuanya dapat diakses jarak jauh melalui internet.

Penulis ingin mengucapkan terima kasih yang tak terhingga kepada beberapa pihak yang telah berperan besar:

- Dr. Ira Puspasari, S.Si, M.T., atas seluruh bimbingan dan arahan yang telah diberikan selama proses penelitian.
- Pimpinan dan staf PT. Meka Prima Solusi atas izin, dukungan, dan peluang yang diberikan untuk melaksanakan penelitian.
- Semua individu yang telah meluangkan waktu dan tenaga untuk membantu penyelesaian laporan ini, baik yang namanya disebutkan maupun yang tidak dapat disebutkan satu per satu.

Penulis menyadari bahwa laporan ini tentu tidak luput dari kekurangan. Oleh karena itu, semua masukan dan kritik konstruktif akan sangat kami hargai demi perbaikan di masa yang akan datang. Kami berharap hasil penelitian ini dapat memberikan manfaat nyata dan menjadi kontribusi positif bagi perkembangan ilmu pengetahuan dan teknologi.

Surabaya, 2025

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Setiap perusahaan, termasuk PT. Meka Prima Solusi, mutlak membutuhkan keamanan yang solid untuk memastikan operasional berjalan lancar dan aset tetap terlindungi. Mengingat perusahaan ini mengelola area kerja dan aset bernilai tinggi, diperlukan sistem pengawasan yang tidak hanya andal, tetapi juga mudah diakses dan real-time. Sistem kamera pengawas (CCTV) analog yang lama seringkali kurang memadai karena akses jarak jauhnya terbatas, pemasangan kabelnya rumit dan mahal, serta informasi keamanannya tersaji lambat. Kendala-kendala ini membuat perusahaan sulit bereaksi cepat jika terjadi insiden atau potensi bahaya.

Dengan adanya kemajuan yang pesat dalam teknologi Internet of Things (IoT) telah membuka jalan bagi solusi keamanan yang jauh lebih modern dan efisien. Menggabungkan *IoT* ke dalam sistem CCTV menghasilkan pengawasan yang lebih hemat biaya dan lebih efektif. Contohnya, dengan menggunakan modul mikrokontroler kecil seperti ESP32-CAM, yang terkenal karena ukurannya yang ringkas, konsumsi daya yang irit, serta adanya Wi-Fi dan kamera bawaan, proses pengembangan sistem pengawasan menjadi jauh lebih sederhana.

Melihat dari kebutuhan dan peluang teknologi tersebut, penelitian ini dilaksanakan untuk merancang dan mengimplementasikan sistem CCTV live stream berbasis *IoT* menggunakan perangkat ESP32-CAM. Tujuannya adalah menghadirkan video streaming langsung yang bisa dipantau dari mana saja, dilengkapi dengan fitur pengambilan foto cepat (snapshot) untuk dokumentasi instan. Harapannya, sistem baru ini akan meningkatkan efektivitas pengawasan dan keamanan di PT. Meka Prima Solusi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana merancang dan mengimplementasikan sistem CCTV berbasis ESP32-CAM yang mampu melakukan *live streaming* video dan pengambilan gambar melalui jaringan internet?
2. Bagaimana kinerja sistem dalam menyajikan *live stream* video dan mengirimkan

gambar *snapshot* yang dapat diakses secara *real-time* oleh pengguna di PT. Meka Prima Solusi?

1.3 Batasan Masalah

Agar penelitian ini lebih terfokus dan tujuan dapat tercapai dengan baik, maka batasan masalah yang ditetapkan adalah sebagai berikut:

1. Perangkat keras utama yang digunakan adalah modul ESP32-CAM sebagai pengolah data dan penangkap gambar.
2. Data yang dikirimkan berupa *live stream* video dan *snapshot* gambar berformat JPEG.
3. Sistem menggunakan koneksi Wi-Fi sebagai media transmisi data (IoT).
4. Fokus penelitian adalah pada implementasi di lingkungan PT. Meka Prima Solusi.
5. Akses pemantauan dilakukan melalui antarmuka *web-based* sederhana yang dapat diakses menggunakan *browser* atau perangkat *smartphone*.

1.4 Tujuan

Tujuan yang ingin dicapai dari penelitian ini adalah:

1. Membangun dan mengimplementasikan sebuah sistem CCTV berbasis ESP32-CAM yang terintegrasi dengan *IoT* untuk menghasilkan *live stream* video dan pengambilan gambar.
2. Menganalisis kinerja sistem dalam hal kecepatan akses dan akurasi pengambilan gambar yang disajikan secara *real-time*.
3. Menghasilkan solusi sistem pemantauan keamanan yang efektif, *cost-effective*, dan mudah diakses untuk PT. Meka Prima Solusi.

1.5 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah:

1. Bagi PT. Meka Prima Solusi: Menyediakan sistem keamanan yang modern, fleksibel, dan terjangkau, sehingga dapat meningkatkan efisiensi pengawasan aset dan lingkungan kerja.
2. Bagi Penulis: Menerapkan pengetahuan teoritis mengenai *IoT*, mikrokontroler (ESP32-CAM), dan pemrograman, serta mendapatkan pengalaman praktis dalam merancang sistem keamanan.

Bagi Akademisi: Dapat menjadi referensi dan dasar pengembangan lebih lanjut dalam

penelitian terkait sistem pengawasan berbasis *IoT* menggunakan *microcontroller* berdaya rendah.

BAB II
GAMBAR UMUM INSTANSI

2.1 Profil Instansi

Gambar 2.1 Logo Perusahaan

PT Meka Prima Solusi adalah perusahaan IT di Surabaya yang fokus pada penyediaan dan integrasi sistem seperti CCTV, jaringan, server, dan kontrol akses untuk mendukung operasional bisnis klien. Gambar 2.1 menunjukkan logo Perusahaan PT.Meka Prima Solusi

2.2 Identitas Perusahaan

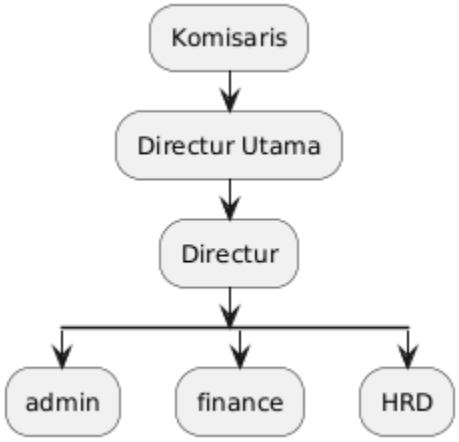
1. Nama Resmi: PT. Meka Prima Solusi.
2. Jenis Badan Hukum: Perseroan Terbatas
3. Alamat: Jl. Wonorejo Permai M/70, Surabaya, Jawa Timur, Indonesia.

2.3 Bidang Usaha & Layanan

1. MPS beroperasi di bidang teknologi informasi, khususnya sebagai system integrator.
2. Layanan yang disebutkan mencakup:
    a. Pemasangan dan integrasi CCTV (pengawasan video)
    b. Jaringan/*Networking*
    c. Sistem PABX (telekomunikasi internal)
    d. Pengadaan dan instalasi server
    e. Kontrol akses (access control) untuk keamanan.
3. Artinya, MPS berfokus pada solusi teknologi untuk infrastruktur IT & keamanan.

2.4 Struktur Organisasi

Pada PT.Meka Prima Solusi yang memiliki kedudukan tertinggi yang teratas adalah komisaris, kedua Direktur Utama, ketiga Direktur, lalu ke admin, finance, dan HRD. Gambar 2.2 menunjukan struktur organisasi pada PT.Meka Prima Solusi.



Gambar 2.2 Struktur Organisasi Perusahaan

BAB III
LANDASAN TEORI

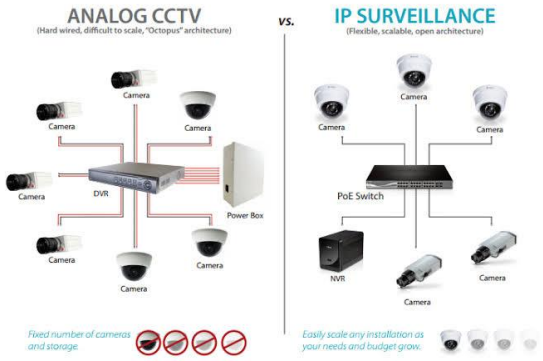
3.1 CCTV

3.1.1 Definisi

Menurut Atmoko (2005) CCTV merupakan instalasi perekaman, seringkali melibatkan minimal satu kamera video, yang fungsi utamanya adalah mengakuisisi dan menghasilkan output berupa data visual maupun audio.(Bahri, 2025)

3.1.2 Komponen Dasar Sistem

Untuk meninjau rekaman terdahulu atau melihat tayangan langsung (*streaming*) dari CCTV, pengguna cukup menggunakan perangkat bergerak (mobile device) asalkan tersambung ke jaringan internet. Agar dapat berfungsi optimal, instalasi CCTV memerlukan beberapa komponen esensial, yaitu kamera (baik model analog maupun berbasis IP), layar pemantau, perangkat perekam (DVR/NVR), dan kelengkapan jaringan komputer. Kamera pengawas harus dipasangkan ke mesin perekam yang relevan—NVR untuk kamera IP atau DVR untuk model analog—baik melalui sambungan kabel atau nirkabel. Khusus untuk koneksi berkabel, kamera analog membutuhkan kabel koaksial, sedangkan kamera IP dihubungkan dengan kabel UTP (*Unshielded Twisted Pair*). Agar fitur *streaming* bisa diakses secara global, perangkat perekam video harus terhubung ke WAN (Wide Area Network) atau internet.(Bata, 2023) Gambar 3.1 menunjukan perbedaan antara CCTV analog dan *IP*



Gambar 3.1 Perbedaan antara CCTV

3.2 Internet Of Things

3.2.1 Definisi

Konsep *Internet of Things* (IoT) merupakan upaya pengembangan fundamental dalam dunia konektivitas. Melalui pemanfaatan sensor, aktuator, dan infrastruktur jaringan, IoT memungkinkan perangkat keras, mesin, hingga objek fisik untuk tersambung sepenuhnya ke internet. Interkoneksi ini menciptakan kemampuan bagi sistem untuk secara mandiri mengumpulkan, memproses data baru, dan selanjutnya mengambil tindakan atau berkolaborasi berdasarkan wawasan yang dihasilkan tersebut.(Nahdi & Dhika, n.d.)

3.2.2 Sejarah

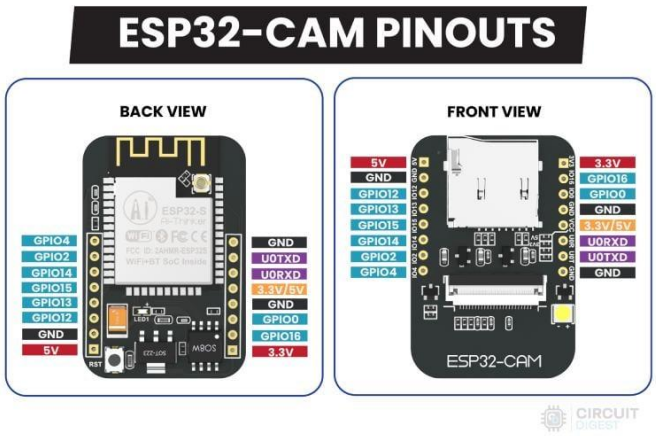
Latar belakang konseptual *Internet of Things* (IoT) dapat ditelusuri kembali ke tahun 1980, ketika para pemrogram di Universitas Carnegie Mellon melakukan terobosan awal. Mereka memodifikasi mesin penjual otomatis Coca-Cola dengan memasang *microswitch* yang terhubung ke internet. Tujuannya ganda: memastikan fungsi pendingin bekerja optimal sehingga minuman tetap dingin, dan memantau ketersediaan stok Coca-Cola secara *real-time*. Aksi inovatif ini dianggap sebagai prekursor yang membuka gerbang bagi kemajuan teknologi di berbagai sektor.

Selanjutnya, pada tahun 1997, Paul Saffo mulai memaparkan potensi besar sensor di masa depan. Namun, terminologi definitif *Internet of Things* (IoT) baru diperkenalkan pada tahun 1999 oleh Kevin Ashton, yang saat itu merupakan anggota Komunitas Radio sekaligus pengembang utama teknologi *Radio Frequency Identification* (RFID) dan menjabat sebagai direktur eksekutif. Seiring dengan pesatnya pertumbuhan perangkat bergerak, komunikasi terpadu, kapabilitas *cloud computing*, dan perluasan konektivitas jaringan, gagasan yang dicetuskan oleh Ashton menjadi semakin penting dan diterima secara mendunia.(Ayu Syahfitri, 2025)

3.2.3 ESP 32 Cam

Sistem pengawasan tertutup (*CCTV*) yang dikembangkan mengadopsi konsep *Internet of Things* (IoT), dengan modul mikrokontroler ESP32-CAM sebagai unit pemroses utama. Perangkat ini dirancang untuk menghasilkan *live feed*

video tanpa komponen audio, di mana pengguna memiliki kapabilitas untuk memantau dan menyimpan *frame* gambar melalui platform aplikasi berbasis *web*. Akses penuh terhadap fungsionalitas dan transmisi data sistem ini memerlukan koneksi internet aktif pada perangkat klien (seperti ponsel pintar atau komputer personal).(Cahyo & Chandra, n.d.) Gambar 3.2 menjelaskan PINOUTS di ESP32 Cam.



Gambar 3.2 ESP32 Cam

1. **Camera OV2460**

Fungsi kamera esp32 ini untuk menangkap gambar atau video. Gambar 3.3 menunjukan komponen *camera* OV2460.

Gambar 3.3 Camera OV2460

2. **USB to TTL Serial Conversion Module**

Perangkat ini berfungsi sebagai jembatan konverter yang esensial, memfasilitasi pertukaran data antara *port Universal Serial Bus* (USB) pada komputer pribadi atau peralatan lainnya, dengan perangkat keras yang beroperasi pada tingkat logika TTL (misalnya, mikrokontroler atau sensor). Secara spesifik, modul ini menyediakan konektivitas serial *Universal Asynchronous Receiver-Transmitter*

(UART) ke USB.(Paramitha & Suartana, 2024) Gambar 3.4 menunjukan modul *conversion USB to TTL.*

Gambar 3.4 Module Conversion USB to TTL

3.3 Live Stream

Live Stream (aliran langsung) adalah proses pengiriman data video dan audio secara *real-time* melalui jaringan. Dalam konteks sistem CCTV berbasis *IoT*, *live stream* merupakan fungsi krusial yang memungkinkan pengguna memantau lokasi secara langsung tanpa jeda waktu yang signifikan.

3.4 Snapshot

Snapshot adalah pengambilan gambar dari ESP32 cam yang dapat di *trigger* secara manual dan secara event contohnya sensor PIR, atau *face recognition*.

BAB IV
PEMBAHASAN

4.1 Analisis Kebutuhan Sistem

Proses analisis ini dilaksanakan dengan tujuan utama mendefinisikan secara cermat persyaratan fungsional dan non-fungsional yang esensial. Hal ini dilakukan demi menjamin bahwa sistem yang akan dikembangkan nantinya mampu beroperasi secara efektif dan sepenuhnya selaras dengan sasaran pengawasan keamanan yang telah ditetapkan oleh PT. MEKA PRIMA SOLUSI.

4.1.1 Kebutuhan Fungsional

Sistem yang dikembangkan harus memenuhi persyaratan fungsional utama berikut:

1. Pemantauan *Real-time*

   Sistem harus menyajikan aliran video langsung tanpa komponen audio dari area yang diawasi, yang dapat diakses melalui

antarmuka berbasis web.

2. Aktivitas Snapshot

   Pengguna harus memiliki kemampuan untuk menangkap dan mengarsipkan frame gambar statis dari live stream ke dalam penyimpanan device melalui web interface.

3. Pengelolaan Data Bukti

   Gambar yang telat diambil (snapshot) harus dapat diunduh langsung oleh pengguna melalui aplikasi web selain kemampuan penyimpanannya

4.1.2 Kebutuhan Non-Fungsional

1. Kinerja

   Dalam hal kinerja sistem, aliran video langsung (*live stream*) harus mencapai laju *frame* minimal 5 FPS (*Frames Per Second*) demi menjamin kualitas pengawasan yang layak dan berkelanjutan. Selain itu, jeda waktu atau *latency* transmisi data video tidak boleh melampaui batas 5 detik, sehingga informasi yang diterima mendekati *real-time*.

2. Keamanan

   Akses terhadap *dashboard* aplikasi berbasis *web* diwajibkan untuk diamankan. Perlindungan ini harus diimplementasikan melalui mekanisme otentikasi yang ketat, seperti prosedur *login* bagi setiap pengguna yang berwenang.

3. Keterjangkauan

   Untuk menjamin efisiensi anggaran dan kepraktisan, seluruh solusi perangkat keras harus dibangun menggunakan komponen yang efisien biaya dan memiliki harga terjangkau, seperti modul mikrokontroler ESP32-CAM.

4.2 Perancangan Sistem

Fase perancangan sistem berperan sebagai langkah fundamental dalam menetapkan arsitektur dan spesifikasi teknis dari solusi CCTV yang memanfaatkan teknologi IoT. Di dalamnya, dilakukan penyusunan desain secara logis maupun fisik

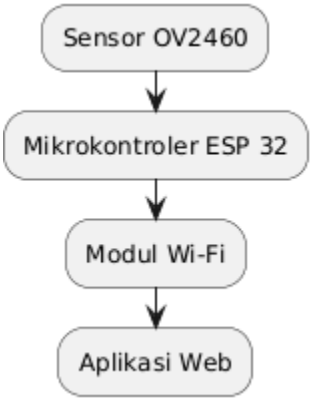
untuk seluruh elemen, mulai dari unit endpoint berbasis ESP32-CAM hingga interface aplikasi web, dengan sasaran akhir mencapai integrasi yang mulus antara seluruh persyaratan fungsional dan non-fungsional yang telah ditetapkan.

4.2.1 Diagram Blok

Diagram blok menunjukkan aliran data dari kamera hingga ke antarmuka pengguna:

- Input: Sensor OV2640 dan Sensor Gerak (opsional).
- Pemroses: Mikrokontroler ESP32 (mengompresi data citra dan menjalankan *web server* lokal).
- Transmisi: Modul Wi-Fi (menghubungkan ke router/internet).
- Output: Aplikasi *Web* (menampilkan *live stream* dan menyimpan data *snapshot*).

Gambar 4.1 menunjukan runtutan kerja sistem.



Gambar 4.1 Diagram Blok Sistem

4.2.2 Perancangan Aplikasi Berbasis Web

Antarmuka *web* dirancang untuk kemudahan penggunaan dan aksesibilitas melalui *smartphone* atau komputer.

1. Halaman Utama: Menampilkan *live feed* video, dan fitur-fitur yang disediakan (seperti resolusi, opsi setting kamera, LED flash, tombol start live stream, dan tombol snapshot).
2. Tombol Kontrol: Terdapat tombol "Capture/Ambil Gambar" yang berfungsi mengirim *command* HTTP POST ke ESP32-CAM untuk

mengambil dan mengirim *frame* gambar.

4.3 Implementasi Sistem

Implementasi mencakup konfigurasi perangkat keras (ESP32-CAM) dan pengembangan perangkat lunak (aplikasi *web*).

4.3.1 Implementasi ESP32

1. Konfigurasi Firmware: Modul ESP32-CAM diprogram menggunakan *Arduino IDE* (atau *Espressif IDF*) dengan *firmware* yang dioptimalkan untuk:
   a. Menginisialisasi OV2640 pada resolusi.

```
config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
   config.fb_count = 2;
#endif
```

   b. Menjalankan *web server* untuk mengirim *live stream* pada alamat IP statis.

```
startCameraServer();

Serial.print("Camera Ready! Use 'http://");
Serial.print(WiFi.localIP());
Serial.println("' to connect");
```

   c. Menunggu *command* untuk fungsi pengambilan gambar.

```
if (config.pixel_format == PIXFORMAT_JPEG) {
  if (psramFound()) {
    config.jpeg_quality = 10;
    config.fb_count = 2;
    config.grab_mode = CAMERA_GRAB_LATEST;
  } else {
    // Limit the frame size when PSRAM is not available
    config.frame_size = FRAMESIZE_SVGA;
```

```
        config.fb_location = CAMERA_FB_IN_DRAM;
      }
    } else {
```

2. Skema Pengkabelan: Penggunaan *converter* diperlukan untuk menghubungkan antarmuka USB komputer ke pin serial UART ESP32-CAM saat proses *upload* program.

```
camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sccb_sda = SIOD_GPIO_NUM;
  config.pin_sccb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.frame_size = FRAMESIZE_UXGA;
  config.pixel_format = PIXFORMAT_JPEG;  // for streaming
```

```
    //config.pixel_format = PIXFORMAT_RGB565; // for face detection/recognition
    config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
    config.fb_location = CAMERA_FB_IN_PSRAM;
    config.jpeg_quality = 12;
    config.fb_count = 1;
```

4.3.2 Implentasi Web Server ESP32

Aktivitas perancangan perangkat lunak pada unit *edge* spesifik, yaitu ESP32-CAM, secara esensial diarahkan untuk mendirikan layanan *server* berbasis protokol HTTP. Fungsionalitas inti dari *server* ini mencakup penanganan transmisi *streaming* secara aktual, eksekusi perintah pengambilan citra tunggal (*snapshot*), serta modifikasi parameter operasional sensor kamera. Pencapaian ini dimungkinkan melalui pemanfaatan *library* ESP-IDF HTTP Server guna membangun kerangka komunikasi yang berdaya guna.

1. Arsitektur Server dan Protokol

   Sistem mengadopsi model arsitektur *dual-server*, yang melibatkan pelaksanaan dua instansi *HTTP server* secara simultan demi mengoptimalkan stabilitas operasional. Pembagian tugas fungsional diatur sebagai berikut: *Server* pertama, dinamakan camera_httpd, didedikasikan untuk mengelola permintaan kendali dan menyajikan konten non-dinamis, mencakup halaman HTML, hasil tangkapan gambar statis (JPEG/BMP), serta *output* status dalam format JSON. Sementara itu, *server* kedua, stream_httpd, secara eksklusif berfokus pada penyaluran *feed* video secara langsung.

```
    httpd_handle_t stream_httpd = NULL;
    httpd_handle_t camera_httpd = NULL;
```

   camera_httpd untuk menangani kontrol dan data statis, sedangankan stream_httpd didirikan pada port yang hanya khusus untuk streaming.

```
    void startCameraServer() {
```

```
  httpd_config_t config = HTTPD_DEFAULT_CONFIG();
  config.max_uri_handlers = 16;

  httpd_uri_t index_uri = {
    .uri = "/",
    .method = HTTP_GET,
    .handler = index_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t status_uri = {
    .uri = "/status",
    .method = HTTP_GET,
    .handler = status_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t cmd_uri = {
    .uri = "/control",
```

```
    .method = HTTP_GET,
    .handler = cmd_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t capture_uri = {
    .uri = "/capture",
    .method = HTTP_GET,
    .handler = capture_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t stream_uri = {
    .uri = "/stream",
    .method = HTTP_GET,
    .handler = stream_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
```

```
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t bmp_uri = {
    .uri = "/bmp",
    .method = HTTP_GET,
    .handler = bmp_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t xclk_uri = {
    .uri = "/xclk",
    .method = HTTP_GET,
    .handler = xclk_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };
```

```
  httpd_uri_t reg_uri = {
    .uri = "/reg",
    .method = HTTP_GET,
    .handler = reg_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t greg_uri = {
    .uri = "/greg",
    .method = HTTP_GET,
    .handler = greg_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t pll_uri = {
    .uri = "/pll",
    .method = HTTP_GET,
    .handler = pll_handler,
```

```
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  httpd_uri_t win_uri = {
    .uri = "/resolution",
    .method = HTTP_GET,
    .handler = win_handler,
    .user_ctx = NULL
#ifdef CONFIG_HTTPD_WS_SUPPORT
    ,
    .is_websocket = true,
    .handle_ws_control_frames = false,
    .supported_subprotocol = NULL
#endif
  };

  ra_filter_init(&ra_filter, 20);

  log_i("Starting web server on port: '%d'", config.server_port);
  if (httpd_start(&camera_httpd, &config) == ESP_OK) {
    httpd_register_uri_handler(camera_httpd, &index_uri);
    httpd_register_uri_handler(camera_httpd, &cmd_uri);
    httpd_register_uri_handler(camera_httpd, &status_uri);
    httpd_register_uri_handler(camera_httpd, &capture_uri);
    httpd_register_uri_handler(camera_httpd, &bmp_uri);
```

```
    httpd_register_uri_handler(camera_httpd, &xclk_uri);
    httpd_register_uri_handler(camera_httpd, &reg_uri);
    httpd_register_uri_handler(camera_httpd, &greg_uri);
    httpd_register_uri_handler(camera_httpd, &pll_uri);
    httpd_register_uri_handler(camera_httpd, &win_uri);
  }

  config.server_port += 1;
  config.ctrl_port += 1;
  log_i("Starting stream server on port: '%d'", config.server_port);
  if (httpd_start(&stream_httpd, &config) == ESP_OK) {
    httpd_register_uri_handler(stream_httpd, &stream_uri);
  }
}
```

Protokol live stream didefinisikan oleh konstanta boundary dan content type yang digunakan dalam stream_handler.

```
#define PART_BOUNDARY “123456789000000000000987654321”
static const char *_STREAM_CONTENT_TYPE = "multipart/x-mixed-replace;boundary=" PART_BOUNDARY;
static const char *_STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";
static const char *_STREAM_PART = "Content-Type: image/jpeg\r\nContent-Length: %u\r\nX-Timestamp: %d.%06d\r\n\r\n";
```

4.3.3 Implementasi Fungsi Kunci

Semua *URI handler* didaftarkan di fungsi startCameraServer(). Berikut implementasi logis dari *handler* kunci:

1. Stream

   Fungsi stream_handler untuk melakukan loop tak terbatas untuk mengambil frame, menformat header, dan mengirimkan chunk

```
static esp_err_t stream_handler(httpd_req_t *req) {
   // ... (deklarasi variabel, inisialisasi header)

   // Pengaktifan LED jika streaming dimulai
#if defined(LED_GPIO_NUM)
   isStreaming = true;
   enable_led(true);
#endif

   while (true) {
      fb = esp_camera_fb_get(); // 1. Ambil frame buffer
      // ... (konversi ke JPEG jika perlu)

      if (res == ESP_OK) {
         res = httpd_resp_send_chunk(req, _STREAM_BOUNDARY,
strlen(_STREAM_BOUNDARY));
      }
      if (res == ESP_OK) {
         // 2. Format header MJPEG (boundary, Content-Length, timestamp)
         size_t hlen = snprintf((char *)part_buf, 128, _STREAM_PART,
_jpg_buf_len, _timestamp.tv_sec, _timestamp.tv_usec);
         res = httpd_resp_send_chunk(req, (const char *)part_buf, hlen);
      }
      if (res == ESP_OK) {
         // 3. Kirim data frame JPEG
         res = httpd_resp_send_chunk(req, (const char *)_jpg_buf, _jpg_buf_len);
      }
      // ... (return frame buffer)

      if (res != ESP_OK) {
         break; // Keluar dari loop jika terjadi kegagalan pengiriman
```

```
        }
        // ... (Perhitungan FPS menggunakan ra_filter_run)
    }

    // Penonaktifan LED jika streaming berhenti
#if defined(LED_GPIO_NUM)
    isStreaming = false;
    enable_led(false);
#endif

    return res;
}
```

2. Capture

   capture_handler menunjukan mekanisme pengambilan gambar (snapshot) dan aktivasi LED sementara.

```
static esp_err_t capture_handler(httpd_req_t *req) {
    // ... (deklarasi variabel)

#if defined(LED_GPIO_NUM)
    enable_led(true); // Hidupkan LED
    vTaskDelay(150 / portTICK_PERIOD_MS); // Tunda sebentar
    fb = esp_camera_fb_get(); // Ambil frame
    enable_led(false); // Matikan LED
#else
    fb = esp_camera_fb_get();
#endif

    // ... (error checking)

    if (fb->format == PIXFORMAT_JPEG) {
```

```
    res = httpd_resp_send(req, (const char *)fb->buf, fb->len); // Kirim JPEG
langsung
  } else {
    // Konversi dan kirim jika format non-JPEG
    jpg_chunking_t jchunk = {req, 0};
    res = frame2jpg_cb(fb, 80, jpg_encode_stream, &jchunk) ? ESP_OK :
ESP_FAIL;
    httpd_resp_send_chunk(req, NULL, 0); // Sinyal akhir chunk
  }
  esp_camera_fb_return(fb);
  // ...
  return res;
}
```

3. Kontrol Parameter

   Fungsi cmd_handler bertanggung jawab memproses permintaan HTTP GET yang datang melalui URI /control. Tujuannya adalah untuk memodifikasi konfigurasi sensor kamera secara *real-time* berdasarkan parameter yang dikirim oleh klien (misalnya, *web interface*).

```
static esp_err_t cmd_handler(httpd_req_t *req) {
  char *buf = NULL;
  char variable[32];
  char value[32];

  if (parse_get(req, &buf) != ESP_OK) {
    return ESP_FAIL;
  }
  // Dapatkan variabel dan nilai dari URL query string
  if (httpd_query_key_value(buf, "var", variable, sizeof(variable)) !=
ESP_OK || httpd_query_key_value(buf, "val", value, sizeof(value)) !=
```

```
        ESP_OK) {
            // ...
            return ESP_FAIL;
        }
        free(buf);

        int val = atoi(value);
        sensor_t *s = esp_camera_sensor_get();
        int res = 0;

        // Contoh pemanggilan setter
        if (!strcmp(variable, "framesize")) {
            res = s->set_framesize(s, (framesize_t)val);
        } else if (!strcmp(variable, "quality")) {
            res = s->set_quality(s, val);
        }
        // ... (banyak parameter lainnya)
    #if defined(LED_GPIO_NUM)
        // Kontrol intensitas LED
        else if (!strcmp(variable, "led_intensity")) {
            led_duty = val;
            if (isStreaming) {
                enable_led(true);
            }
        }
    #endif
        // ...
        return httpd_resp_send(req, NULL, 0);
}
```

4. Status kamera

   status_handler membangun respons JSON.

```
static esp_err_t status_handler(httpd_req_t *req) {
    static char json_response[1024];

    sensor_t *s = esp_camera_sensor_get();
    char *p = json_response;
    *p++ = '{';

    // ... (Pengambilan register spesifik sensor - OV5640/OV2640)

    // Pengambilan dan penulisan status umum sensor ke string JSON
    p += sprintf(p, "\"xclk\":%u,", s->xclk_freq_hz / 1000000);
    p += sprintf(p, "\"pixformat\":%u,", s->pixformat);
    p += sprintf(p, "\"framesize\":%u,", s->status.framesize);
    p += sprintf(p, "\"quality\":%u,", s->status.quality);
    p += sprintf(p, "\"brightness\":%d,", s->status.brightness);
    // ... (parameter status lainnya)

#if defined(LED_GPIO_NUM)
    p += sprintf(p, ",\"led_intensity\":%u", led_duty);
#else
    p += sprintf(p, ",\"led_intensity\":%d", -1);
#endif
    *p++ = '}';
    *p++ = 0;

    httpd_resp_set_type(req, "application/json");
    httpd_resp_set_hdr(req, "Access-Control-Allow-Origin", "*");
    return httpd_resp_send(req, json_response, strlen(json_response));
}
```

4.4 Analisis Kinerja Sistem

Resolusi	Ukuran Bingkai Rata-rata (B)	Waktu Kompresi Rata-rata (ms)	FPS Rata-rata (AVG)	Status Kinerja
320x240(QVGA)	4000	20	41.7	Sangat Optimal
400x296(CIF)	5766	20.5	41.7	Sangat Optimal
640x480(VGA)	10700	105	9.5	Kurang Optimal (Limit kinerja)
800x600(SVGA)	22000	142	7	Tidak Optimal

Tabel 4.1 Analisis Kinerja Sistem

1. Zona Kinerja Optimal (Resolusi Rendah)

   Resolusi 320x240 dan 400x296 berada di dalam batas kemampuan perangkat keras Anda.

   a. Kecepatan Luar Biasa: Kedua resolusi ini mempertahankan waktu kompresi yang sangat rendah rata-rata 20 ms, menghasilkan FPS Rata-rata yang tinggi dan stabil rata-rata 41.7 fps.

   b. Titik Terbaik *(Sweet Spot)*: 400x296 adalah pilihan terbaik. Meskipun memiliki 54% lebih banyak piksel daripada 320x240, peningkatan ukuran bingkai rata-rata 44% berhasil dikelola oleh CPU tanpa perlambatan, memberikan kualitas gambar yang lebih baik dengan kecepatan yang sama.

2. Zona Kegagalan Kinerja (*Bottleneck*)

   Peningkatan ke 640x480 dan 800x600 menunjukkan bahwa perangkat keras Anda mencapai batas komputasi dan memicu *bottleneck*.

   a. 640x480 (Titik Batas): Peningkatan ke 640x480 menyebabkan ukuran bingkai melebihi 10.000 Byte. Hal ini membuat waktu pemrosesan melonjak tinggi (ke rata-rata 105ms), sehingga FPS Rata-rata anjlok ke rata-rata 9.5 fps.

   b. 800x600 (Kegagalan Parah): Peningkatan resolusi lebih lanjut hingga 800x600 (ukuran bingkai rata-rata 22.000 Byte) memperparah masalah. Waktu pemrosesan semakin tidak stabil dan FPS rata-rata turun hingga rata-rata 7 fps.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan seluruh proses implementasi dan pengujian sistem *live streaming* kamera ESP32-CAM, dapat disimpulkan bahwa tujuan utama proyek telah tercapai dengan keberhasilan implementasi arsitektur *dual server* HTTP yang secara efektif memisahkan tugas kontrol dan *streaming*, sehingga mampu menjaga stabilitas dan efisiensi kinerja secara keseluruhan. Penggunaan protokol MJPEG (*Motion JPEG*) di atas HTTP terbukti andal dalam menyediakan *feed* video dinamis ke klien *web browser*. Verifikasi kinerja menunjukkan bahwa sistem memenuhi persyaratan waktu nyata berkat integrasi Running *Average Filter*, yang memberikan metrik Frames Per Second (FPS) rata-rata yang stabil dan akurat. Selain itu, cmd_handler telah berhasil menyediakan antarmuka yang fleksibel untuk kontrol parameter sensor secara *real-time* (seperti kualitas dan resolusi), sementara fitur *snapshot* dengan aktivasi LED terprogram meningkatkan kualitas pengambilan gambar statis. Secara keseluruhan, sistem ini berhasil mewujudkan solusi *monitoring* berbasis IoT yang fungsional, stabil, dan dapat dikonfigurasi melalui antarmuka web.

5.2 Saran

Untuk pengembangan dan peningkatan kualitas sistem di masa depan, diajukan beberapa saran teknis:

1. Migrasi ke Protokol *Streaming* yang Lebih Ringan: Meskipun MJPEG efektif, disarankan untuk menjajaki implementasi *streaming* berbasis protokol yang lebih hemat *bandwidth*, seperti H.264 atau H.265 melalui WebSocket atau RTMP, dengan memanfaatkan kemampuan akselerasi video pada *chipset* ESP32 yang lebih baru (seperti ESP32-S3), guna mendukung resolusi dan *framerate* yang lebih tinggi.
2. Manajemen Alokasi Memori Adaptif: Mengembangkan mekanisme *firmware* untuk secara otomatis menyesuaikan alokasi *frame buffer* atau menurunkan kualitas (*quality*) dan resolusi (*framesize*) secara progresif apabila mendeteksi tekanan memori (*heap* kritis) atau *load* CPU yang berlebihan.
3. Penerapan Keamanan Jaringan: Menambahkan lapisan keamanan pada *web*

server, minimal berupa Autentikasi Dasar HTTP pada *endpoint* kontrol (/control) dan *streaming* untuk mencegah akses tidak sah, mengingat ESP32-CAM sering digunakan dalam lingkungan jaringan lokal.

4. Optimasi Konsumsi Daya: Menambahkan mode tidur (*deep sleep*) yang dapat dipicu melalui *web interface* atau timer. Fungsi ini penting untuk aplikasi berbasis baterai, di mana ESP32 dapat bangun hanya untuk mengambil gambar atau melakukan *streaming* singkat, lalu kembali ke mode hemat daya.
5. Pengembangan *Endpoint* Kontrol Lanjutan: Memperluas fungsi *endpoint* /status untuk tidak hanya menampilkan status sensor, tetapi juga metrik kesehatan sistem secara keseluruhan (suhu CPU, penggunaan *heap* memori), dikirim dalam format JSON yang terstruktur.

DAFTAR PUSAKA

Ayu Syahfitri. (2025). Internet of Things (IoT), Sejarah, Teknologi, dan Penerapannya. *Uranus : Jurnal Ilmiah Teknik Elektro, Sains Dan Informatika*, *3*(1), 113–120. https://doi.org/10.61132/uranus.v3i1.667

Bahri, S. (2025). Analisa Pemilihan CCTV (Closed Circuit Television) Sebagai Alat Keamanan Untuk Penggunaan di Perumahan & Perkantoran. *Jurnal Minfo Polgan*, *14*(1), 537–541. https://doi.org/10.33395/jmp.v14i1.14788

Bata, E. S. (2023). UTILIZATION OF CCTV ANALOG CAMERA ON MOBILE WEB-BASED ROOM SECURITY MONITORING SYSTEM. *Jurnal Komputer Dan Informatika*, *11*(1), 29–36. https://doi.org/10.35508/jicon.v11i1.10033

Cahyo, E. N., & Chandra, Z. R. (n.d.). *RANCANG BANGUN APLIKASI CCTV BERBASIS INTERNET OF THINGS*. *29*(2), 2686–4711. https://doi.org/10.36309/goi.v29i2.201

Nahdi, F., & Dhika, H. (n.d.). *Analisis Dampak Internet of Things (IoT) Pada Perkembangan Teknologi di Masa Yang Akan Datang 33*.

Paramitha, P., & Suartana, I. M. (2024). Analisis QoS Dan QoE Pada Video Streaming Berbasis IoT Menggunakan ESP32-CAM dan NGROK. *Journal of Informatics and Computer Science*, *05*.