BAB I

PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi yang sangat pesat membuat pemerintah tidak cukup hanya menjadi regulator namun juga perlu menjadi fasilitator ataupun akselerator di era industri 4.0 ini. Salah satunya adalah pengembangan ekosistem *Internet of Things* (IoT).

Seiring dengan perkembangan zaman dan perkembangan teknologi yang semakin pesat, tingkat kejahatan semakin meningkat pula. Era globalisasi telah menuntut manusia untuk menciptakan keamanan dalam bekerja. Demikian halnya dengan sistem keamanan gedung untuk perkantoran maupun rumah mewah lainnya juga ikut berkembang. Sistem keamanan yang ada antara lain menggunakan kamera CCTV melalui operator yang meminta. Perkembangan teknologi CCTV telah dapat ditranformasikan menjadi file digital yang dapat disimpan dalam media penyimpanan dan dapat dilhat melalui internet. Sekarang CCTV dapat diconver dengan jaringan dan diaplikasikan dengan teknologi lainnya. Kamera CCTV merupakan sistem keamanan yang paling banyak digunakan saat ini. Data informasi kejadian yang dapat direkam secara visual melalui media penyimpan dan dapat diputar kembali dalam bentuk video. Penerapan sistem keamanan di ruangan Lab Informatika melalui pemanfaatan CCTV merupakan suatu upaya untuk memonitoring keadaan ruangan. Pada prinsipnya sistem CCTV yang umum digunakan saat ini adalah dengan

menggunakan perangkat kamera sebagai media pemantau, yang terkoneksi dengan perangkat PC yang difungsikan sebagai media penyimpanan dan monitoring hasil pemantauan dari kamera. Kelemahan pada sistem ini adalah dalam hal melakukan monitoring hasil dari rekaman vidio tidak tersimpan

CCTV (Closed Circuit Television) adalah sistem pengawasan atau monitoring suatu kawasan menggunakan kamera video yang dipasang ditempat-ditempat tertentu, dirangkai menjadi sebuah jaringan tertutup dan dapat dipantau dari sebuah ruang kontrol. Di sisi lain, Selain itu adanya koneksi internet yang terjangkau semakin membuat perangkat ini seolah-olah menjadi barang yang harus dimiliki. Pada era kemajuan teknologi informasi saat ini kebutuhan akan informasi dan komunikasi yang cepat dan *up-to date* sangatlah dibutuhkan untuk menunjang aktivitas pekerjaan sehari-hari. Dengan adanya alat CCTV mengunakan *ESP32-CAM* berbasis *Internet of Things* bisa meminilisir tingkat kejahatan karena kejadian akan terekam dalam cctv sehingga dapat dilihat kejadian sebelum melalui kamera CCTV.

Dari latar belakang yang telah dijelaskan diatas, maka penulis memilih judul **“RANCANG BANGUN MEMONITORING RUANGAN LAB INFORMATIKA MENGGUNAKAN *ESP32-CAM* BERBASIS INTERNET OF THINGS”**

B. Rumusan Masalah

Adapun yang menjadi rumusan masalah dalam penelitian ini adalah : bagaimana merancang dan membangun CCTV mengunakan *ESP32-CAM* berbasis *Internet Of Things*?

C. Batasan masalah

Adapun yang menjadi batasan masalah dalam penelitian ini adalah :

1. Sistem yang di buat masih berbentuk prototipe
2. Memonitoring ruangan Lab Informatika menggunakan menggunakan 1 kamera *ESP32-CAM*.

D. Tujuan Penelitian

Adapun tujuan yang hendak dicapai dalam penyusunan ini:

1. Merancang dan membuat elektronika prangkat CCTV menggunakan *ESP32-CAM* berbasis *internet of things.*
2. Membuat perangkat untuk memantau keamanan ruangan Lab informatika menggunakan *ESP32-CAM* berbasis *internet of things.*

E. Manfaat penelitian

1. Manfaat bagi penulis.

   Agar penulis dapat mengaplikasikan ilmu dan teori yang didapat di bangku perkuliahan, serta dapat menambah pengetahuan tentang *Internet of Things* (IoT) secara mendalam.

2. Manfaat bagi masyarakat.

   Diharapkan dapat menjadi *Alternatif* bagi masyarakat agar selalu tetap waspada akan kejahatan.

F. Sistematika Penulisan

Dalam menyusun sistematika penulisan, penulis menguraikan ke dalam lima bab yaitu :

BAB I : PENDAHULUAN

Menguraikan tentang latar belakang penulisan, rumusan masalah, tujuan penelitian, manfaat penelitian, dan batasan masalah, dan juga sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Pada Bab ini berisi uraian - uraian tentang tinjauan penelitian sebelumnya dan teori - teori yang digunakan sebagai referensi dalam rancang bangun memonitoring ruangan lab informatika menggunakan esp32-cam berbasis internet of things.

BAB III : METODE PENELITIAN

Membahas tentang waktu dan tempat penelitian, metode penelitian, jenis penelitian, alat dan bahan yang digunakan dalam penelitian serta desain sistem.

BAB IV : HASIL DAN PEMBAHASAN

Bab ini membahas tentang Analisis Sistem Dengan UML, Perancangan aplikasi,Implementasi, dan Pengujian sistem.

BAB V : PENUTUP

Membahas tentang kesimpulan dan saran.

BAB II

TINJAUAN PUSTAKA

A. Penelitian Sebelumnya

Pengertian penelitian adalah suatu proses penyelidikan yang dilakukan secara aktif, tekun, dan sistematis, dimana tujuannya untuk menemukan, menginterpretasikan, dan merevisi fakta-fakta.

1. Hasil penelitian yang pernah dilakukan Dedi Setiawan (2019) Universitas Putera Batam, dengan judul penelitian "Perancangan sistem pengontrol keamanan rumah dengan smart cctv mengunakan arduino berbasis telegram" pada penelitian ini pengujian alat dengan aplikasi telegram untuk megontrol alat.
2. Dwi Ely Kurniawan dan Syafaruddi Fani (2017) Politeknik negeri batam melakukan penelitian dengan judul "Perancangan system kamera pengawas berbasis perangkat bergerak mengunakan raspberry pi". Pada penelitian ini peneliti menggunakan raspberry pi sebagai microcontrollernya.
3. Okkita Rizan dan Hamidah (2016) STMIK Atma Luhur Pangkalpinang. Melakukan penilitian dengan judul "Rancangan aplikasi monitoring kamera cctv untuk perangkat mobile berbasis android ". Rancangan fitur aplikasi ini mampu streaming video, merecording video, dan menampilkan image kapan saja mereka mau melihatnya pada perangkat mobile berbasis android.

B. Konsep Dasar Sistem

a. Pengertian Sistem

Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu (Jogiyanto, 2015). Sistem ini menggambarkan suatu kejadian-kejadian dan kesatuan yang nyata, seperti tempat, benda dan orang-orang yang betul-betul ada dan terjadi.

Sistem dapat didefinisikan sekumpulan elemen yang saling terkait dan terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem.

b. Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat - sifat yang tertentu yaitu :

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi atau saling bekerjasama untuk membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

2. Batas Sistem

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas suatu sistem menunjukkan ruang lingkup dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem yang bersifat menguntungkan dan juga dapat bersifat merugikan sistem tersebut.

4. Penghubung Sistem

   Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Keluaran *(output)* dari satu subsistem akan menjadi masukan *(input)* untuk subsistem yang lainnya dengan melalui penghubung.

5. Masukan Sistem

   Masukan *(input)* adalah energi yang dimasukkan kedalam sistem, dapat berupa masukan perawatan *(Maintenance Input)* dan masukan sinyal *(Signal Input).*

6. Keluaran Sistem

   Keluaran *(output)* adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolah Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolahan yang akan merubah masukan menjadi keluaran.

8. Sasaran Sistem

Suatu sistem mempunyai tujuan atau sasaran, sasaran dari sistem sangat menentukan masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. (Kadir Abdul.2015).

C. Sistem Monitoring

Sistem adalah sekumpulan elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan dan merupakan sekumpulan komponen yang saling bekerjasama untuk mencapai tujuan guna memperbaiki organisasi kearah yang lebih baik.

Banyak definisi dalam menjelaskan pengertian sistem. Namun demikian, pada umumnya definisi itu menggambarkan bahwa pengertian sistem mengandung dua konotasi, yaitu benda atau entitas, dan proses atau metode. Schrode dan Voich (1974) dalam bukunya yang berjudul *organization and management: basic systems concept* misalnya, menyatakan bahwa sistem adalah "*whole compounded of several parts"* (suatu kesatuan yang tersusun dari sejumlah elemen.

Monitoring adalah proses pengumpulan informasi mengenai apa yang sebenarnya terjadi selama proses implementasi atau penerapan program. Monitoring adalah proses rutin pengumpulan data dan pengukuran kemajuan atas objek program/memantau perubahan yang fokus pada proses dan keluaran.

Menurut Edi Suharto monitoring pada dasarnya adalah pemantauan suatu kegiatan proyek atau program sosial yang dilaksanakan pada saat kegiatan tersebut sedang berlangsung.

Menurut Nalahudin Muhlisin monitoring merupakan suatu proses pengumpulan dan menganalisis informasi dari penerapan suatu program termasuk mengecek secara regular untuk melihat apakah kegiatan atau program itu berjalan sesuai dengan rencana sehingga masalah yang dilihat atau ditemui dapat diatasi. Monitoring menyediakan data dasar untuk menjawab permasalahan.

Dari beberapa pendapat diatas dapat dipahami pengertian monitoring adalah pemantauan suatu program yang dilaksanakan saat kegiatan tersebut sedang berlangsung, monitoring menyediakan data dasar untuk menjawab permasalahan.

Sedangkan yang dimaksud dengan system monitoring adalah layanan yang melakukan proses pengumpulan data dan melakukan analisis terhadap data-data tersebut dengan tujuan untuk memaksimalkan seluruh sumber daya yang dimiliki.

D. Pengertian Perancangan

Menurut Ahmad Shukri Mohd (2015:5) perancangan sistem adalah sebuah teknik pemecahan masalah yang saling melengkapi (dengan analisis sistem) yang merangkai kembali bagian-bagian komponen yang menjadi sistem yang lengkap. Hal ini melibatkan penambahan, penghapusan dan perubahan-perubahan bagian relative pada sistem awal (aslinya) yang tidak didukung oleh ketiga pilar ini tidak

dapat dikatakan sebagai informasi yang berguna, tetapi merupakan sampah (*garbage*).

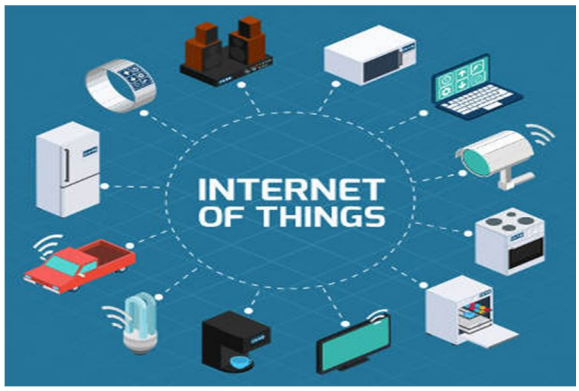
Sistem informasi adalah cara-cara yang terorganisasi untuk mengumpulkan, memasukkan, dan mengolah serta menyimpan data, dan cara-cara yang diorganisasi untuk menyimpan, mengelolah, mengendalikan, dan melaporkan informasi sedemikian rupa sehingga sebuah organisasi dapat mencapai tujuan yang telah ditetapkan. (Krismaji, 2015).

E. *Internet of Things* (IoT)

Perkembangan teknologi semakin pesat dari waktu ke waktu. Mulai dari mobil pintar *(smart car)* yang bisa berjalan sendiri ke berbagai tujuan tanpa pengemudi manusia, hingga perangkat rumah pintar *(smart home)* semacam Alexa yang bisa otomatis bersuara mengingatkan untuk melakukan aktivitas sesuai jadwal. Seluruh teknologi terbaru ini adalah bagian dari *Internet of Things. Internet of Things (IoT)* adalah sebuah konsep di mana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan tanpa memerlukan adanya interaksi dari manusia ke manusia atau dari manusia ke komputer. *Internet of Things (IoT)* adalah struktur di mana objek, orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke komputer. (Burange & Misalkar, 2015).

IoT sudah berkembang pesat mulai dari penggabungan teknologi nirkabel, *Micro Electromechanical Systems (MEMS)* dan juga Internet. *Internet of Things (IoT)* menggunakan beberapa teknologi yang secara garis besar digabungkan menjadi satu kesatuan diantaranya sensor sebagai pembaca data, koneksi *internet* dengan beberapa macam topologi jaringan, Radio *Frequency Identification (RFID), wireless sensor network* dan teknologi yang terus akan bertambah sesuai dengan kebutuhan. *Internet of Things (IoT)* juga bisa mencakup teknologi-teknologi sensor lainnya, seperti teknologi nirkabel maupun kode QR yang sering kita temukan di sekitar kita, contoh penerapannya dalam benda yang ada di dunia nyata adalah untuk pengolahan bahan pangan, elektronik, dan berbagai mesin atau teknologi lainnya yang semuanya tersambung ke jaringan lokal maupun global lewat sensor yang tertanam dan selalu menyala aktif. *Internet of Things (IoT)* ini mengacu pada mesin atau alat yang bisa diidentifikasikan sebagai representasi virtual dalam strukturnya yang berbasis *Internet*.

Tantangan terbesar yang bisa menjadi hambatan dalam mengkonfigurasi *Internet of Things (IoT)* adalah menjembatani kesenjangan antara dunia fisik dan dunia informasi dan bagaimana menyusun jaringan komunikasinya, dikarenakan jaringan yang dibutuhkan oleh IoT sangatlah kompleks. Selain itu, *Internet of Things (IoT)* juga memerlukan suatu sistem keamanan yang cukup ketat. Di samping masalah tersebut, biaya pengembangan *Internet of Things (IoT)* yang mahal juga sering menjadi faktor penyebab kegagalan, sehingga pembuatan dan pengembangannya bisa berakhir gagal produksi berakhir gagal produksi.



Gambar. 2.I. Cara kerja Internet of Things (IoT)
(Sumber: www.dewaweb.com)

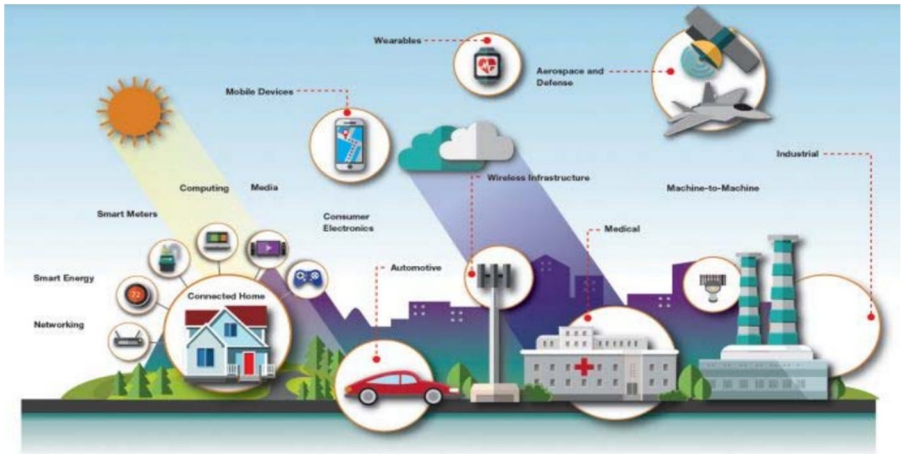
Cara kerja *Internet of Things (IoT)*, dengan memanfaatkan suatu argumentasi *pemrograman*, di mana tiap - tiap perintah argumen tersebut dapat menghasilkan suatu interaksi antar mesin yang telah terhubung secara otomatis tanpa campur tangan manusia dan tanpa dibatasi oleh jarak yang jauh. *Internet* menjadi penghubung antara kedua interaksi mesin tersebut. Manusia dalam *Internet of Things (IoT)* tugasnya hanyalah menjadi pengatur dan pengawas dari mesin-mesin yang bekerja secara langsung tersebut.

Unsur-unsur pembentuk IoT yang mendasar adalah:

1. Kecerdasan Buatan *(Artificial Intelligence*/AI), *Internet of Things (IoT)* membuat hampir semua mesin yang ada menjadi *"Smart"* (pintar). Ini berarti *Internet of Things (IoT)* bisa meningkatkan segala aspek kehidupan kita dengan pengembangan teknologi yang didasarkan pada AI. Pengembangan teknologi yang ada dilakukan dengan pengumpulan data, algoritma kecerdasan buatan, dan jaringan yang tersedia. Contohnya sederhana seperti meningkatkan atau mengembangkan perangkat lemari

es/kulkas sehingga dapat mendeteksi jika stok susu dan sereal sudah hampir habis, bahkan bisa juga membuat pesanan ke supermarket secara otomatis jika stok akan habis.

2. Konektivitas dalam *Internet of Things (IoT)*, ada kemungkinan untuk membuat atau membuka jaringan baru, dan jaringan khusus IoT. Jaringan ini tidak lagi terikat hanya dengan penyedia utamanya saja. Jaringannya tidak harus berskala besar dan mahal, bisa tersedia pada skala yang jauh lebih kecil dan lebih murah. *Internet of Things (IoT)* bisa menciptakan jaringan kecil di antara perangkat sistem.
3. Sensor merupakan pembeda yang membuat *Internet of Things (IoT)* unik dibanding mesin canggih lainnya. Sensor ini mampu mendefinisikan instrumen, yang mengubah *Internet of Things (IoT)* dari jaringan standar dan cenderung pasif dalam perangkat, sehingga menjadi suatu sistem aktif yang dapat diintegrasikan ke dunia nyata dalam kehidupan sehari-hari.
4. Keterlibatan Aktif *(Active Engagement), Internet of Things (IoT)* mengenalkan paradigma yang baru bagi konten aktif, produk, maupun keterlibatan layanan.
5. Perangkat Berukuran Kecil. *Internet of Things (IoT)* memanfaatkan perangkat-perangkat kecil yang dibuat khusus agar menghasilkan ketepatan, skalabilitas, dan fleksibilitas yang baik.



Gambar. 2.2. Penerapan Internet of Things (IoT)
(Sumber: www.dewaweb.com)

Berbagai macam penerapan *Internet of Things (IoT)* dalam beberapa sektor antara lain:

1. Pertanian

   Beberapa diantaranya seperti mengumpulkan data soal suhu, curah hujan, kelembaban, kecepatan angin, serangan hama, dan muatan tanah. Data-data tersebut bisa dipakai untuk mengotomatisasi teknik pertanian dan dapat dipakai untuk mengambil keputusan *(decision making)* berdasarkan informasi yang diperoleh sehingga dapat meningkatkan kualitas dan kuantitas, meminimalkan risiko dan limbah, serta mengurangi upaya yang diperlukan dalam mengelola tanaman. Sebagai contoh, petani sekarang sudah bisa memantau suhu dan kelembaban tanah dari jauh, dan bahkan menerapkan data yang diperoleh *Internet of Things (IoT)* untuk program pemupukan yang lebih merata.

2. Energi

Sejumlah perangkat yang memakan energi besar saat ini sudah bisa terintegrasi dengan konektivitas *internet* yang memungkinkan mesin-mesin maupun jaringan untuk berkomunikasi dalam menyeimbangkan pembangkitan listrik serta penggunaan energi yang lebih hemat dan efektif. Perangkat ini juga bisa memungkinkan akses *remote control* dari pengguna. Selain itu, bisa juga mengaktifkan fungsi semacam penjadwalan (misalnya untuk menyalakan/mematikan mesin pemanas, mengendalikan oven, mengubah kondisi pencahayaan dari terang menjadi redup hingga ke gelap, dan lain sebagainya).

3. Lingkungan

Aplikasi pemantauan lingkungan dari *Internet of Things (IoT)* biasanya menggunakan sensor dalam membantu terwujudnya perlindungan lingkungan. Contoh penerapannya seperti pemantauan kualitas udara atau air, kondisi atmosfer atau tanah, dan dapat mencakup pemantauan terhadap satwa liar dan habitatnya. *Internet of Things (IoT)* juga dimanfaatkan dalam penanggulangan bencana seperti sistem peringatan dini Tsunami atau gempa bumi. Perangkat *Internet of Things (IoT)* dalam hal ini memiliki jangkauan geografis yang sangat luas serta mampu bergerak.

4. Otomatisasi Rumah

Perangkat *Internet of Things (IoT)* dimanfaatkan untuk memantau dan mengontrol sistem mekanis dan elektronik yang digunakan pada berbagai

jenis bangunan, seperti industri atau perumahan. Selain itu juga dapat mengontrol penggunaan energi secara *realtime* dalam mengurangi konsumsi energi. Perangkat *Internet of Things (IoT)* dapat terintegrasi menjadi sistem rumah pintar. Contoh penerapannya seperti: lampu menyala secara otomatis ketika malam hari, kemudian lampu akan mati secara otomatis pada saat jadwal tidur. Pagi hari, taman Anda akan disiram air oleh mesin penyiram otomatis. Begitu juga dengan kulkas Anda yang bisa memesan stok makanan sendiri ketika habis.

5. Medik dan Kesehatan

   Dalam dunia medik dan kesehatan, Perangkat *Internet of Things (IoT)* akan merekam data-data kesehatan pasien dan ditransfer langsung ke tenaga medis maupun Rumah Sakit. Data-data yang bisa dideteksi dan dikirimkan seperti detak jantung, tingkat gula dalam darah, dan lain sebagainya. *Smartphone* atau ponsel akan jadi alat pemantau kesehatan yang canggih dan tentunya bisa sangat membantu. Salah satu contoh penerapannya seperti, tempat tidur pintar yang bisa otomatis memberitahukan dokter atau perawat ketika pasien hendak bangun dari tempat tidur.

6. Transportasi

   Penerapan *Internet of Things (IoT)* dapat membantu manusia dalam berkomunikasi, mengontrol, dan pemrosesan informasi pada berbagai aspek sistem transportasi. Interaksi dinamis yang terjadi antara komponen-

komponen itu berasal dari sebuah sistem transportasi. Sistem tersebut memungkinkan komunikasi antara kendaraan dengan kontrol lalu lintas yang lebih efektif, parkir yang lebih cerdas, manajemen logistik dan armada, kontrol kendaraan, dan juga terkait faktor keselamatan maupun bantuan di jalan raya.

F. Mikrokontroler

Mikrokontroler adalah mikroprosesor yang dikhususkan untuk instrumentasi dan kendali. Mikrokontroler merupakan suatu alat elektronika *digital* yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus (Zarkasyi, 2013).

Mikrokontroler adalah sebuah sistem komputer lengkap dalam satu serpih (*chip*) yang biasanya digunakan untuk sebuah *embedded system* (sistem yang dibentuk guna menjalankan satu atau lebih dari suatu fungsi tertentu secara *real time*). Mikrokontroler biasanya berukuran kecil karena didesain hanya untuk satu fungsi tertentu pada suatu sistem. Pemanfaatan mikrokontroler umumnya digunakan di bidang kendali dan instrumentasi elektronik. Mikrokontroler lebih dari sekedar sebuah mikroprosesor karena sudah terdapat atau berisikan ROM (*Read-Only Memory)*, RAM (*Read-Write Memory*), beberapa Port masukan maupun keluaran, dan beberapa peripheral seperti pencacah/pewaktu, ADC (*Analog to Digital converter*), DAC (*Digital to Analog converter*) dan *serial* komunikasi.

Setiap mikrokontroler memiliki arsitektur yang berbeda - beda tergantung perancangannya. Meskipun demikian, setiap arsitektur mikrokontroler pada dasarnya memiliki keseragaman pada pokok-pokok dan cara kerjanya. Berdasarkan arsitektur, mikrokontroler terbagi dua yaitu:

1. CISC (*Complex Instruction Set Computing)*

   Complex Instruction Set Computing (CISC) atau kumpulan instruksi komputasi kompleks. Adalah suatu arsitektur komputer di mana setiap instruksi akan menjalankan beberapa operasi tingkat rendah, seperti pengambilan dari memori (*load*), operasi aritmatika, dan penyimpanan ke dalam memori (*store*) yang saling bekerja sama.

2. RISC *(Reduced Instruction Set Computer)*

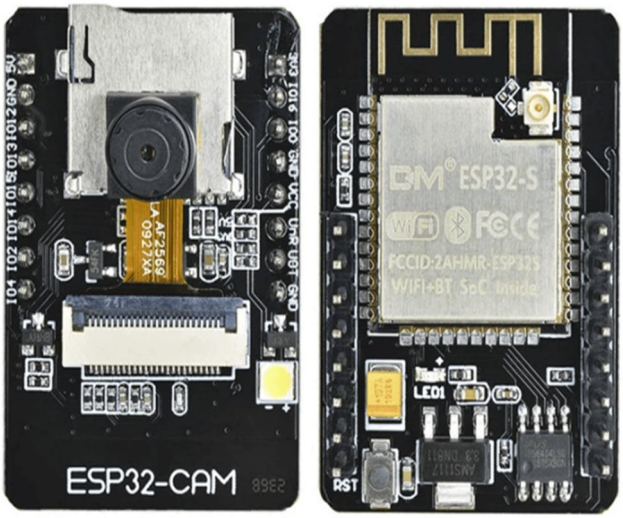
   Reduced Instruction Set Computer (RISC). Merupakan bagian dari arsitektur mikroprosessor, berbentuk kecil dan berfungsi untuk negeset instruksi dalam komunikasi diantara arsitektur yang lainnya (Setiawan, 2006).

G. Esp32 CAM

ESP32 CAM merupakan salah satu mikrokontroler yang memiliki fasilitas tambahan berupa *bluetooth*, *wifi*, kamera, bahkan sampai ke *slot mikroSD*. *ESP32 CAM* ini biasanya digunakan untuk project IoT *(Internet of Things)* yang membutuhkan fitur kamera. Modul *ESP32 CAM* memiliki lebih sedikit pin I/O dibandingkan modul ESP32 produk sebelumnya, yaitu *ESP32 Wroom*. Hal ini dikarenakan sudah banyak pin yang digunakan secara internal untuk fungsi kamera dan fungsi *slot* kartu *microSD*. Selain itu, modul *ESP32 CAM* juga tidak

memiliki port USB khusus (mengirim program dari *port USB* komputer). Jadi untuk memprogram modul ini Anda harus menggunakan *USB TTL* atau kita dapat menambahkan modul tambahan berupa downloader khusus untuk *ESP32 CAM*.

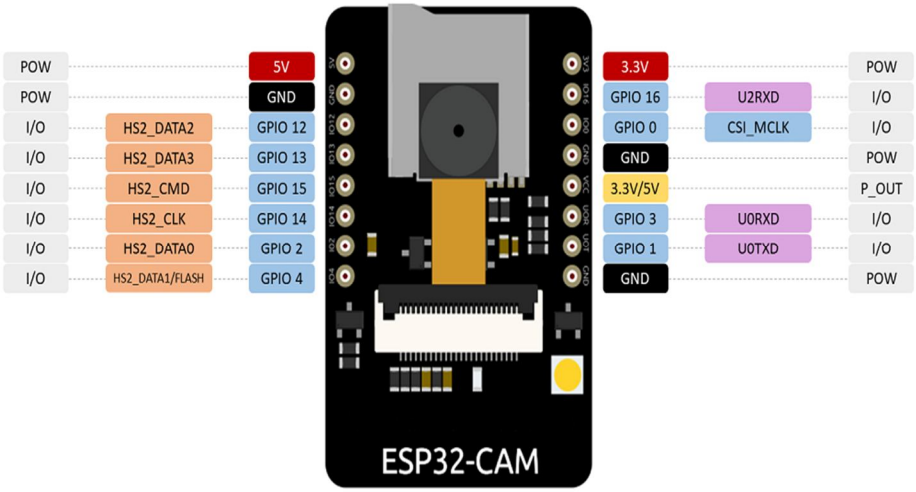
Modul *ESP32 CAM* memiliki 2 sisi dalam rangkaian modulnya. Di bagian atas terdapat modul kamera yang dapat dibongkar pasang dan ada *microSD* yang dapat diisi, serta flash sebagai lampu tambahan untuk kamera jika diperlukan. Di bagian belakang modul, terdapat antena internal, konektor untuk antena eksternal, pin male untuk I/O dan *ESP32S* sebagai otaknya. Lebih jelasnya, kita dapat melihat spesifikasinya sebagai berikut:



Gambar 2.3 *ESP32 CAM*
Sumber (http://www.labelektronika.com)

Tabel 2. 1 Spesifikasi *ESP32 CAM*

Module Model	***ESP32CAM***
Package	DIP-16
Size	27*40.5*4.5 (±0.2) mm
SPI Flash	*Default* 32*Mbit*
RAM	*Internal* 520*KB + External* 4*M PSRAM*
Bluetooth	*Bluetooth* 4.2 *BR/EDR and BLE standard*
Wi-fi	802.11 *b/g/n/e/i*
Support Interface	*UART, SPI, I2C, PWM*
Support TF Card	*Maximum support 4G*
IO Port	9
Serial Port Rate	*Default 115200 bps*
Image Output Format	*JPEG (Only OV*2640 *support), BMP, GRAYSCALE*
Spectrum Range	2412 ~ 2484MHz
Antenna Form	*Onboard PCB antenna, gain 2dBi*
Transmit Power	802.11b: 17 ± 2 *dBm (@*11*Mbps)* 802.11g: 14 ± 2 *dBm (@*54*Mbps)* 802.11n: 13 ± 2 *dBm (@MCS7)*
Safety	*WPA/WPA2/WPA2-Enterprise/WPS*
Power Supply Range	*5V*
Operating Temperature	-20 ℃ ~ 85 ℃
Storage Environment	-40 ℃~ 90℃, < 90%RH



Gambar 2.4 pinout *ESP32 CAM*
Sumber (http://www.labelektronika.com)

Module *ESP32 CAM* juga sudah di lengkapi dengan *GPIO* yang dapat di program sebagai Input dan Output , untuk lebih deteil Fitur atau Fungsi pin Khusus sebagai Berikut :

Power Pins

Module ESP32 CAM memiliki 3 pin *GND*, bisa di lihat di gambar yang warna hitam. dan untuk *Power* ada 2 opsi bisa 5V atau 3.3 V yang pin warna merah, jadi Development Board *ESP32 CAM* bisa menggunakan 5V atau 3.3 V, tapi kalau saran dari penulis lebih baik pakai yang 5V supaya lebih stabil.

Power Output Pin

Untuk Output Pin memiliki 1 pin, bisa 3.3V atau 5V bisa di set di *jumper* di *boardnya*, kalau pengalaman penulis *defaultnya* 3.3V

Serial Pins

Module Development Board ESP32 CAM ini Sudah memiliki *Serial Hardware* di pin *GPIO1* untuk *U0TXD* dan pin *GPIO3* untuk *U0RXD*, Kita dapat

menggunakan pin ini untuk komunikasi dengan sensor atau *module2* yang menggunakan komunikasi *serial*.

Tapi *ESP32 CAM* tidak memiliki built-in *Programmer*, sehingga kita membutuhkan *pin serial* ini untuk digunakan *upload* atau *flash* program ke *ESP32 CAM*, menggunakan *Serial Programmer* seperti *FTDI Module*.

GPIO 0

Pin GPIO 0 di gunakan untuk *Flash Mode Selection*, jadi *PIn GPIO 0* digunakan untuk mensetting *ESP32 CAM* dalam *mode upload* atau *Flash* program ketika terhubung ke *GND*, saat selesei di *upload* , *GPIO* 0 di lepas dari pin *GND* supaya mode nya kembali ke sistem normal, jadi system bisa jalan normal keluar dari mode *flash* atau upload. jadi pin ini kita hubungkan dengan *GND* ketika kita mau upload atau *Flash* aja.

Module Micro SD Card Reader

ESP32 CAM sudah terintegrasi dengan *Module Micro SD*, dapat di fungsikan untuk di pakai untuk menyimpan gambar atau data logger lainnya.

untuk pin yang di pakai deteilnya sebagai berikut :

Tabel 2.2 Pin *ESP32 CAM* untuk menyimpan gambar atau data

Module Micro SD	ESP32
CLK	GPIO 14
CMD	GPIO 15
DATA0	GPIO 2
DATA1	GPIO 4
DATA2	GPIO 12
DATA3	GPIO 13

Module Kamera

Untuk *Module* Camera *ESP32 CAM* sudah terintegrasi dengan *FPC* Camera *Connector* 24 *Pin* dan di paket penjualan sudah dapat Modul *OV2640* untuk cameranya, sehingga tinggal di psang di connector FPC Cameranya, sehingga Module Camera *ESP32 CAM* sudah siap di pakai sudah terintegrasi dengan *module* Camera *OV2640*. Berikut Koneksi Kamera Dengan *ESP32 CAM AI-Thinker:*

Tabel 2.3 *Pin* Koneksi Kamera Dengan *ESP32 CAM*

OV2640	**ESP32**
D0	GPIO 5
D1	GPIO 18
D2	GPIO 19
D3	GPIO 21
D4	GPIO 36
D5	GPIO 39
D5	GPIO 34
D5	GPIO 35
XCLK	GPIO 0
PCLK	GPIO 22
SYNC	GPIO 25
HREF	GPIO 23
SDA	GPIO 26
SCL	GPIO 27
POWER PIN	GPIO 32

GPIO 4

Pin ini merupakan Built-in Flash, Pin ini bisa di gunakan untuk Flash Light camera atau fungsi lainnya

GPIO 33

ESP32_CAM memiliki Built-in yang sudah terintegrasi di modulenya, jadi bisa di gunakan untuk test program Blink atau untuk test status program.

H. Kabel *Jumper*

(Razor, 2020) Kabel *Jumper* menjadi salah satu komponen yang sangat penting dalam membuat rangkaian. Kabel *Jumper* adalah kabel elektrik yang memiliki pin konektor di setiap ujungnya untuk menghubungkan dua komponen tanpa harus memerlukan solder. Konektor pada kabel *Jumper* terdapat dua jenis yaitu konektor jantan (*male connector*) dan konektor betina (*female connector*). Ukuran panjang kebel *Jumper* juga bervarian. Seperti yang diketahui bahwa kabel *Jumper* memiliki beberapa warna yang berbeda, perbedaan warna tidak berpengaruh pada fungsi kabel perbedaan warna tersebut hanya untuk memudahkan saat membuat rangkaian. Pada pembuatan Rancang Bangun Memonitoring Ruangan Lab Informatika Menggunakan *Esp32Cam* Berbasis *Internet Of Things*, kabel *Jumper* yang digunakan adalah kabel *Jumper* jenis *Male to Female* dan *Male to Male*,

1. ***Male to Male***

Kabel jumper jenis ini digunakan untuk koneksi male to male pada kedua ujung kabelnya.



Gambar 2.9 Kabel *JumperMale to Female*
(Sumber : http://repository.untag-sby.ac.id)

2. ***Male to Female***

Kabel jumper jenis ini digunakan untuk koneksi male to female dengan salah satu ujung kabel dikoneksi male dan satu ujungnya lagi dengan koneksi female.



Gambar 2.10 Kabel *Jumper Male to Female*
(Sumber : http://repository.untag-sby.ac.id)

3. *Female to Female*

Kabel jumper jenis ini digunakan untuk koneksi female to female pada kedua ujung kabelnya.



Gambar 2.11 Kabel *Female to Female*
(Sumber : http://repository.untag-sby.ac.id)

Tabel 2. 5 Spesifikasi Kabel *Jumper*

Jenis	*Male To Female*
Panjang Kabel	±10 Cm
Ukuran *Pitch*	2.54 mm

I. Software Arduino

Diperlukan software Arduino IDE (*Integrated Development Environment)* untuk menulis program pada board Arduino. Arduino IDE adalah sebuah software yang digunakan untuk menulis bahasa program lalu mengompilasinya menjadi bilangan biner yang selanjutnya diupload ke dalam memori arduino ataupun mikrokontroler.

1. **Menu-menu yang ada pada software Arduino IDE**

Verify

Verifiy yang tersedia pada Arduino IDE digunakan sebagai pemeriksaan kode program yang telah kami buat. Apakah sudah benar atau belum, jika belum nanti akan muncul sebuah message box yang memberitahukannya.

Upload

Menu upload idgunakan sebagai untuk untuk mengupload sketch yang telah selesai ke board Arduino. Jika menggunakan menu upload maka akan otomatis Arduino IDE akan melakukan sebuah verify lebih dulu sebelum melakukan proses upload.

New

Menu new berfungsi sebagai membuat halaman sketch baru.

Open

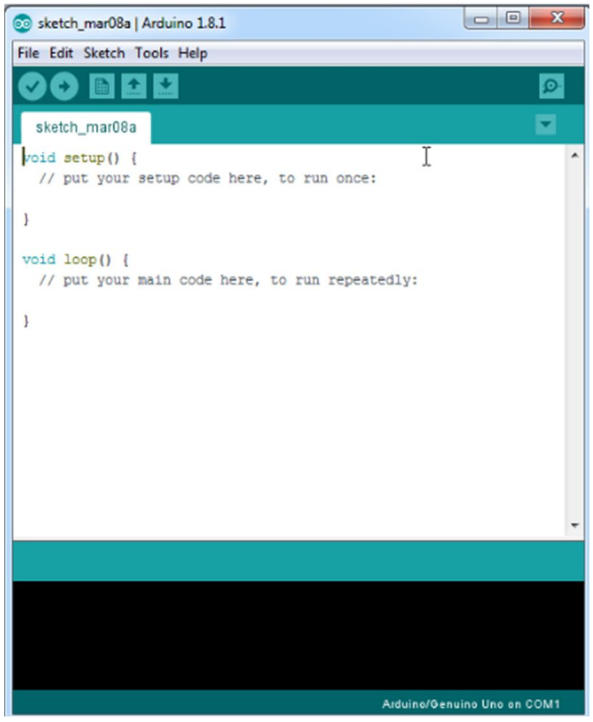
Menu open fungsinya digunakan untuk membuat proyek sketch yang sudah tersimpan di dalam folder sebelumnya.

Save

Menu save berguna untuk melakukan penyimpanan sketch yang sudah dibuat.

Serial Monitor

Menu serial monitor digunakan untuk membuka serial monitor untuk menampilkan sebuah jendela yang dikirimkan atau dipertukarkan antara skecth dan Arduino pada port serialnya.



Gambar 2.12 Software Arduino IDE
(Sumber : www.sinauarduino.com)

2. Fungsi Arduino IDE

Fungsi Arduino IDE (Integrated Development Environment) dapat menulis sketch. Sketch ditulis dalam suato editor sebuah teks dan disimpan dengan ekstensi .ino. Teks editor yang ada pada Arduino IDE yang memiliki fitur untuk cut and paste dan search and replace sehingga programmer dapat lebih mudah dalam melakukan penulisan kode program.

Tidak hanya itu saja software Arduino IDE juga mempunyai fitur message box, yang mempunyai warna hitam guna menampilkan sebuah status seperti pesan error, compile, dan melakukan program. Di bagian paling kanan dari software ini juga menunjukkan sebuah board yang terkonfigurasi sekaligus COM Ports yang mau digunakan.

3. Bahasa Pemprograman Arduino Berbasis Bahasa C

Seperti yang telah dijelaskan diatas program *Arduino* sendiri menggunakan bahasa C. walaupun banyak sekali terdapat bahasa pemrograman tingkat tinggi (*high level language*) seperti *pascal*, *basic*, cobol, dan lainnya. Walaupun demikian, sebagian besar dari paraprogramer profesional masih tetap memilih bahasa C sebagai bahasa yang lebih unggul, berikut alasan-alasannya:

a) Bahasa C merupakan bahasa yang *powerful* dan *fleksibel* yang telah terbukti dapat menyelesaikan program-program besar seperti pembuatan sistem operasi, pengolah gambar (seperti pembuatan game) dan juga pembuatan kompilator bahasa pemrograman baru.

b) Bahasa C merupakan bahasa yang *portabel* sehingga dapat dijalankan di beberapa sistem operasi yang berbeda. Sebagai contoh program yang kita tulis dalam sistem operasi *windows* dapat kita kompilasi didalam sistem operasi *linux* dengan sedikit ataupun tanpa perubahan sama sekali.

c) Bahasa C merupakan bahasa yang sangat populer dan banyak digunakan oleh programer berpengalaman sehingga kemungkinan besar *library* pemrograman telah banyak disediakan oelh pihak luar/lain dan dapat diperoleh dengan mudah.

d) Bahasa C merupakan bahasa yang bersifat modular, yaitu tersusun atas rutin-rutin tertentu yang dinamakan dengan fungsi *(function*) dan fungsi-fungsi tersebut dapat digunakan kembali untuk pembuatan program-program lainnya tanpa harus menulis ulang implementasinya.

e) Bahasa C merupakan bahasa tingkat menengah (*middle level language*) sehingga mudah untuk melakukan interface (pembuatan program antar muka) ke perangkat keras.

f) Struktur penulisan program dalam bahasa C harus memiliki fungsi utama, yang bernama main. Fungsi inilah yang akan dipanggil pertama kali pada saat proses eksekusi program. Artinya apabila kita mempunyai fungsi lain selain fungsi utama, maka fungsi lain tersebut baru akan dipanggil pada saat digunakan.

Oleh karena itu bahasa C merupakan bahasa prosedural yang menerapakan konsep runtutan (program dieksekusi per baris dari atas ke bawah secara berurutan), maka apabila kita menuliskan fungsi-fungsi lain tersebut dibawah fungsi utama, maka kita harus menuliskan bagian prototipe (*prototype*), hal ini dimaksudkan untuk mengenalkan terlebih dahulu kepada kompiler daftar fungsi yang akan digunakan di dalam program. Namun

apabila kita menuliskan fungsi-fungsi lain tersebut diatas atau sebelum fungsi utama, maka kita tidak perlu lagi untuk menuliskan bagian prototipe diatas(Djuandi, Feri. (2015).

Selain itu juga dalam bahasa C kita akan mengenal *file header,* biasa ditulis dengan ekstensi h(*.h), adalah file bantuan yang yang digunakan untuk menyimpan daftar-daftar fungsi yang akan digunakan dalam program. Bagi anda yang sebelumnya pernah mempelajari bahasa pascal, *file header* ini serupa dengan unit. Dalam bahasa C, *file header* standar yang untuk proses *input/output* adalah <*stdio.h*>.

Perlu sekali untuk diperhatikan bahwa apabila kita menggunakan *file header* yang telah disediakan oleh kompilator, maka kita harus menuliskannya didalam tanda‘<’ dan ‘>’ (misalnya <*stdio.h*>). Namun apabila menggunakan *file header* yang kita buat sendiri, maka file tersebut ditulis diantara tanda “ dan ” (misalnya “coba *header.h*”). perbedaan antara keduanya terletakpada saat pencerian file tersebut. Apabila kita menggunakan tanda <>, maka file tersebut dianggap berada pada direktori deafault yang telah ditentukan oleh kompilator. Sedangkan apabila kita menggunakan tanda “”, maka *file header* dapat kita dapat tentukan sendiri lokasinya.

File header yang akan kita gunakan harus kita daftarkan dengan menggunakan directive *#include.* Directive *#include* ini berfungsi untuk memberi tahu kepada kompilator bahwa program yang kita buat akan menggunakan *file-file* yang didaftarkan.

a) *Power Arduino* dapat diberikan *power* melalui koneksi *USB* atau *power supply*. *Power*nya diselek secara otomatis. *Power supply* dapat menggunakan adaptor DC atau baterai. Adaptor dapat dikoneksikan dengan mencolok *jack* adaptor pada koneksi *port input supply*. *Board arduino* dapat dioperasikan menggunakan *supply* dari luar sebesar 6 - 20 volt. Jika *supply* kurang dari 7V, kadangkala pin 5V akan menyuplai kurang dari 5 *volt* dan *board* bisa menjadi tidak stabil. Jika menggunakan lebih dari 12 V, tegangan di regulator bisa menjadi sangat panas dan menyebabkan kerusakan pada *board*. Rekomendasi tegangan ada pada 7 sampai 12 volt.

b) Vin Tegangan *input* ke *board arduino* ketika menggunakan tegangan dari luar (seperti yang disebutkan 5 *volt* dari koneksi *USB* atau tegangan yang diregulasikan). Pengguna dapat memberikan tegangan melalui pin ini, atau jika tegangan suplai menggunakan *power jack*, aksesnya menggunakan pin ini.

c) 5V Regulasi *power supply* digunakan untuk *power mikrokontroller* dan komponen lainnya pada board. 5V dapat melalui Vin menggunakan regulator pada *board*, atau *supply* oleh *USB* atau *supply* regulasi 5V lainnya.

d) 3V3 *Supply* 3.3 *volt* didapat oleh FTDI *chip* yang ada di *board*. Arus maximumnya adalah 50mA

e) Pin *Ground* berfungsi sebagai jalur *ground* pada *arduino*

f) Memori *ATmega328* memiliki 32 KB *flash* memori untuk menyimpan kode, juga 2 KB yang digunakan untuk *bootloader*. *ATmega328* memiliki 2 KB untuk SRAM dan 1 KB untuk EEPROM.

Input & *Output* Setiap 14 pin digital pada *arduino* dapat digunakan sebagai *input* atau *output*, menggunakan fungsi pin *Mode*(), digital *Write*(), dan digital *Read*(). *Input/output* dioperasikan pada 5 *volt*. Setiap pin dapat menghasilkan atau menerima maximum 40 mA dan memiliki *internal pull-up* resistor (*disconnected* oleh *default*) 20-50K Ohm.

J. Pengertian *Website* dan HTTP

1. Definisi *Website*

Sebuah situs *web* (sering pula disingkat menjadi situs saja*; website*, *site*) adalah sebutan bagi sekelompok halaman *web* (*webpage*), yang umumnya merupakan bagian dari suatu nama *domain* (*domain name*) atau *sub domain* di *world wide web* (*WWW*) di *internet*. WWW terdiri dari seluruh situs *web* yang tersedia kepada publik. Halaman-halaman pada sebuah situs *web* diakses dari sebuah *URL* yang menjadi "akar" (*root*), yang disebut *homepage* (halaman induk; sering diterjemahkan menjadi "beranda", "halaman muka"), dan biasanya disimpan dalam *server* yang sama. Tidak semua situs *web* dapat diakses dengan gratis. Beberapa situs *web* memerlukan pembayaran agar dapat menjadi pelanggan, misalnya situs-situs yang menampilkan situs-situs berita, layanan surat elektronik (*email*), dan lain-lain.

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu komponen-komponen Sistem (*Sistem Components*), batasan-batasan sistem (*Sistem Boundaries*), penghubung sistem (*Sistem Interface*), lingkungan luar sistem (*Sistem Environment*), masukan sistem (*Sistem Input*), pengolah sistem (*Sistem Process*), keluaran sistem (*Sistem Output*), dan sasaran sistem (*Sistem Target*).

Menurut Hidayat dalam Daud F. Tatang (2013), "*Website* atau situs dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman."

Menurut Kustiyahningsih dalam Daud F. Tatang (2013), "*Web* adalah layanan yang didapat oleh pemakai komputer yang terhubung ke *internet*"

Browser adalah perangkat lunak untuk mengakses halaman-halaman *web* seperti *Internet Explorer, Mozilla Firefox, Opera, Safari, Google Chrome, Rock Melt*, dan lain-lain. Terminologi *website* adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah *domain* atau *sub domain*, yang tempatnya berada di dalam *World Wide Web (WWW)* di *internet*. Sebuah *web page* adalah dokumen yang ditulis dalam format *HTML (Hyper Text Markup Language*), yang hampir selalu bisa diakses melalui *HTTP*, yaitu protokol yang menyampaikan informasi dari *server website* untuk ditampilkan kepada para

pemakai melalui *web browser*. Semua publikasi dari *website-website* tersebut dapat membentuk sebuah jaringan informasi yang sangat besar.

Halaman-halaman dari *website* akan bisa diakses melalui sebuah URL yang biasa disebut *homepage*. *URL* ini mengatur halaman-halaman situs untuk menjadi sebuah hirarki, meskipun *hyperlink-hyperlink* yang ada di halaman tersebut mengatur para pembaca dan memberitahu mereka susunan keseluruhan dan bagaimana arus informasi ini berjalan. Beberapa *website* membutuhkan sub skripsi (data masukan/*subscript*) agar para *user* bisa mengakses sebagian atau keseluruhan isi *website* tersebut. Contohnya ada beberapa *situs-situs* bisnis, *situs-situs email* gratisan, yang membutuhkan *sub* skripsi agar kita bisa mengakses *situs-situs* tersebut.

Penemu *website* adalah Sir Timothy John "Tim" Berners Lee, sedangkan *website* yang tersambung dengan jaringan, pertama kalinya muncul pada tahun 1991. Maksud dari Tim ketika membuat *website* adalah untuk mempermudah tukar menukar dan memperbarui informasi kepada sesama peneliti di tempat dia bekerja. Pada tanggal 30 April 1993, CERN (tempat dimana Tim bekerja) menginformasikan bahwa WWW dapat digunakan secara gratis oleh semua orang.

Sebuah *website* bisa berupa hasil kerja dari perorangan atau individu, atau menunjukkan kepemilikan dari sebuah organisasi, perusahaan, dan biasanya *website* itu menunjukkan beberapa topik khusus, atau kepentingan tertentu. Sebuah *website* bisa berisi *hyperlink* yang menghubungkan ke *website* lain. Jadi,

kadangkala perbedaan antara *website* yang dibuat oleh individu perseorangan dengan *website* yang dibuat oleh organisasi bisnis bisa saja tidak kentara.

Website ditulis, atau secara dinamik di konversi menjadi *HTML* dan akses melalui sebuah program *software* yang biasa disebut dengan *web browser*, yang dikenal juga dengan HTTP *Client*. Halaman *web* dapat dilihat atau diakses melalui jaringan komputer dan *internet*, perangkat bisa saja berupa Personal *Computer*, Laptop *Computer*, PDA atau *Cellphone*.

Sebuah *website* dibuat di dalam sebuah sistem komputer yang dikenal dengan *Web server*, juga disebut HTTP *server*, dan pengertian ini juga bisa menunjuk pada *software* yang dipakai untuk menjalankan sistem ini, yang kemudian menerima lalu mengirimkan halaman-halaman yang diperlukan untuk merespon permintaan dari *user*. *Apache* adalah bahasa *program/software* yang biasa digunakan didalam sebuah *web server*, kemudian setelah itu adalah *Microsoft Internet Information Server* (IIS).

Sebuah *website* statik, adalah salah satu bentuk *website* yang isi didalam *website* tersebut tidak dimaksudkan untuk di *update* secara berkala, dan biasanya diperbaiki secara manual oleh beberapa orang yang menggunakan *software editor*. Ada 3 tipe kategori *software editor* yang biasa dipakai untuk tujuan memperbaiki ini, mereka adalah:

1) Elemen 1, *Text* Editor;. Contohnya adalah *Notepad* atau Edit, dimana HTML diubah didalam program *editor* tersebut.

2) Elemen 2, WYSIWYG (*What You See Is What You Get*) Editor. Contohnya *Microsoft Frontpage* dan *Macromedia Dreamweaver*, dimana *situs* di *edit* menggunakan GUI (*Graphic User Interface*) dan format HTML secara otomatis di *generate* oleh editor ini.
3) Elemen 3 yaitu editor yang sudah memiliki *template*, contohnya *Rapidweaver* dan *iWeb*, dimana, editor ini memperbolehkan *user* untuk membuat dan mengupdate *website*-nya langsung ke *web server* secara cepat, tanpa harus mengetahui apapun tentang HTML. Mereka dapat memilih template yang sesuai dengan keinginan mereka, menambah gambar atau objek, mengisinya dengan tulisan dan dengan sekejap mereka sudah dapat membuat *website* tanpa harus melihat sama sekali kode-kode HTML.

Sebuah *website dynamic* adalah *website* yang secara berkala, informasi didalamnya berubah, atau *website* ini bisa berhubungan dengan user dengan berbagai macam cara atau metode, (HTTP *cookies* atau variabel *database*, sejarah kunjungan, variabel sesi, dan lain-lain) bisa juga dengan cara interaksi langsung menggunakan *form* dan pergerakan *mouse*. Ketika *web server* menerima permintaan dari *user* untuk memberikan halaman tertentu, maka halaman tersebut akan secara otomatis di ambil dari media penyimpanan sebagai respon dari permintaan yang diminta oleh *user*. Sebuah situs dapat menampilkan dialog yang sedang berlangsung diantara dua *user*, memantau perubahan situasi, atau menyediakan informasi yang berkaitan dengan sang *user*.

Ada banyak sistem *software* yang dapat dipakai untuk *menggenerate dynamic web system dan situs dynamic*, beberapa diantaranya adalah *Coldfusion* (CFM), *Active Server Pages* (ASP), *Java Server Pages* (JSP) dan PHP, bahasa program yang mampu untuk meng-generate *dynamic web system* dan situs dinamis. Situs juga bisa termasuk didalamnya berisi informasi yang diambil dari satu atau lebih *database* atau bisa juga menggunakan teknologi berbasis XML, contohnya adalah RRS. Isi situs yang statis juga secara periodik di *generate*, atau, apabila ada keadaan dimana dia butuh untuk dikembalikan kepada keadaan semula, maka dia akan di *generate*, hal ini untuk menghindari kinerja supaya tetap terjaga.

Plugin atau dapat disebut pula *extension* tersedia untuk menambah banyaknya *feature* dan melengkapi kemampuan dari *web browser*, dimana plugin ini dipakai untuk membuka *content* yang biasanya berupa cuplikan dari gambar bergerak *(active content)* contohnya adalah *flash*, *shockwave* atau *applets* yang ditulis dalam bahasa JAVA. *Dynamic* HTML juga menyediakan untuk *user* supaya dia bisa secara *interaktif* dan *realtime*, *mengupdate* di *web page* tersebut (catatan; halaman yang dirubah, tak perlu di *load* atau di *reload* agar perubahannya dapat dilihat), biasanya perubahan yang dilakukan mereka memakai DOM dan *javascript* yang sudah tersedia pada semua *web browser* sekarang ini.

Seperti yang ditulis diatas, di luar sana ada beberapa perbedaan dalam penulisan dari terminology *website*. Walaupun *website* sudah secara umum dipakai, namun untuk *associated press stylebook, reuters, microsoft, academia*,

dan kamus-kamus yang ada, penulisan yang mereka pakai adalah dengan menggunakan 2 kata, yaitu *website*. Hal ini karena *web* bukanlah terminologi umum, namun ia adalah kependekan dari *world wide web*.

2. Definisi HTTP

HTTP (*HyperText Transfer Protocol*) adalah protokol yang digunakan untuk mentransfer dokumen dalam *World Wide Web* (*WWW*). Protokol ini adalah protokol ringan, tidak berstatus dan generik yang dapat dipergunakan berbagai macam tipe dokumen.

Pengembangan *HTTP* dikoordinasi oleh *World Wide Web* Consortium (W3C) dan grup bekerja *Internet Engineering Task Force* (IETF), bekerja dalam publikasi satu seri *Request For Comment (RFC),* yang paling terkenal RFC 2616, yang menjelaskan HTTP/1.1, versi HTTP yang digunakan umum sekarang ini.

HTTP adalah sebuah protokol meminta/menjawab antara *Client* dan *server* sebuah *client* HTTP seperti *web browser*, biasanya memulai permintaan dengan membuat hubungan TCP/IP ke *port* tertentu di tuan rumah yang jauh (biasanya *port* 80). Sebuah *server* HTTP yang mendengarkan di *port* tersebut menunggu client mengirim kode permintaan (*request*), seperti "*Get* / HTTP/1.1" (yang akan meminta halaman yang sudah ditentukan), diikuti dengan pesan MIME yang memiliki beberapa informasi kode kepala yang menjelaskan aspek dari permintaan tersebut, diikuti dengan badan dari data tersebut. Beberapa kepala (*header*) juga bebas ditulis atau tidak, sementara lainnya (seperti tuan rumah) diperlukan oleh HTTP/1.1. Begitu menerima kode permintaan (dan pesan, bila

ada), *server* mengirim kembali kode jawaban, seperti “200 ok”, dan sebuah pesan yang diminta, atau sebuah pesan *error* atau pesan lainnya.

HTTP pertama kali dipergunakan dalam WWW pada tahun 1990. Pada saat tersebut yang dipakai adalah HTTP *versi* 0.9. *Versi* 0.9 ini adalah protokol transfer dokumen secara mentah, maksudnya adalah data dokumen dikirim sesuai dengan isi dari dokumen tersebut tanpa memandang tipe dari dokumen.

Kemudian pada tahun 1996 HTTP diperbaiki menjadi HTTP versi 1.0. Perubahan ini untuk mengakomodasi tipe-tipe yang hendak dikirim beserta *encoding* yang digunakan dalam pengiriman data dokumen.

Sesuai dengan perkembangan infrastruktur *internet* maka pada tahun 1999 dikeluarkan HTTP versi 1.1 untuk mengakomodasi *proxy, cache, koneksi persistent.*

L. Web Server

Web adalah tampilan pada browser dengan alamat domain khusus untuk sistem penelitian ini. Web dapat dibangun dengan menggunakan bahasa HTML dan PHP dengan style tampilan menggunakan bahasa CSS. Web tersebut disimpan pada satu komputer yang disebut server. Server menyimpan program web dan database untuk dapat diakses oleh admin atau client dari browser. Website dapat dibangun menggunakan program notepad/notepad++ atau adobe dreasweaver.

Web server adalah sebuah software yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada client yang dikenal dan biasanya kita kenal dengan nama web browser dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman web dan pada umumnya akan berbentuk dokumen HTML. Dalam bentuk sederhana web server akan mengirim data HTML kepada permintaan web browser sehingga akan terlihat seperti pada umumnya yaitu sebuah tampilan website.

Fungsi utama web server adalah untuk melakukan atau akan tranfer berkas permintaan pengguna melalui protokol komunikasi yang telah ditentukan sedemikian rupa. Halaman web yang diminta terdiri dari berkas teks, video, gambar, file dan banyak lagi. pemanfaatan web server berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman web termasuk yang di dalam berupa teks, video, gambar atau banyak lagi.

Beberapa Jenis Web Server di antanya adalah:

1. Apache Web Server / The HTTP Web Server
2. Apache Tomcat
3. Microsoft windows Server 2008 IIS (Internet Information Services)
4. Lighttpd
5. Zeus Web Server
6. Sun Java System Web Server

Dari daftar list diatas yang paling dominan dan digunakan hanya pada Apache dan Microsoft Windows Server.

M. UML (Unified Modelling Language)

UML (*Unified Modelling Language*) menurut menurut Adi nugroho (2010:6), "*Unified Modelling Language* adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek". Pemodelan (*modeling*) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

Metode *Unified Modelling Language* (UML) menggunakan tiga bangunan dasar untuk mendeskripsikan sistem atau perangkat lunak yang akan dikembangkan, yaitu:

1. Sesuatu (*things*)

   Ada empat *things* dalam *Unified Modelling Language* (UML):

   a. *Structural things*, bagian yang relatif statis dapat berupa elemen-elemen yang bersifat fisik maupun konseptual.
   b. *Behavioral things,* bagian dinamis biasanya merupakan kata kerja dari model UML yang mencerminkan perilaku sepanjang waktu
   c. *Grouping things,* bagian pengorganisasian dalam UML. Dalam penggambaran model UML yang rumit diperlukan penggambaran paket yang menyederhanakan model. Paket-paket ini kemudian dapat didekomposisi lebih lanjut. Paket berguna bagi pengelompokan sesuatu, misalnya model-model serta subsistem-subsistem.

d. *Notational things,* merupakan bagian yang memperjelas model UML. Dapat berisi komentar yang menjelaskan fungsi serta ciri-ciri tiap elemen.

2. Relasi (*relationship*)

Ada empat *relationship* (hubungan) dalam *Unified Modelling Language* (UML):

a. Ketergantungan (d*ependency*) adalah hubungan dimana perubahan yang terjadi pada suatu elemen independent akan mempengaruhi elemen yang bergantung padanya.

b. Asosiasi adalah apa dan bagaimana yang menghubungkan antara objek satu dengan yang lainnya. Suatu bentuk asosiasi adalah agregasi yang menampilkan hubungan suatu objek dengan bagian-bagiannya.

c. Generalisasi adalah hubungan dimana objek anak berbagai perilaku dan struktur data dari objek yang ada diatasnya (objek induk). Arah dari objek induk ke objek anak dinamakan spesialisasi sedangkan arah sebaliknya dinamakan generalisasi..

d. Realisasi adalah operasi yang benar-benar dilakukan oleh suatu objek.

3. Diagram, *Unified Modelling Language* (UML) menyediakan Sembilan jenis diagram yang dapat dikelompokkan berdasarkan sifatnya (statis dan dinamis).

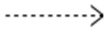
a. diagram *use case*

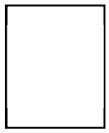
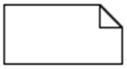
Diagram *use case* menyajikan interaksi antara *use case* dan aktor, dimana aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang di bangu. *Use case*

menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai. (*Sholiq, 2006*)

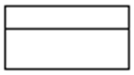
Adapun simbol-simbol *Use Case* Diagram antara lain :

Tabel 2.6 Simbol Use Case Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
NO	**GAMBAR**	**NAMA**	**KETERANGAN**
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

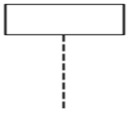
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Tabel 2. 7 Simbol Class Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.

4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri

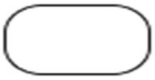
Tabel 2. 8 Simbol Sequence Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
NO	**GAMBAR**	**NAMA**	**KETERANGAN**
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi

Tabel 2. 9 Simbol StateChart Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Tabel 2. 10 Simbol Activity Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

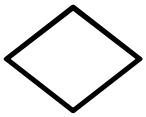
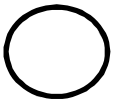
N. *Flowchart*

Flowchart adalah adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses (*instruksi*) dengan proses lainnya dalam suatu program.

Dalam perancangan *flowchart* sebenarnya tidak ada rumus atau patokan yang bersifat mutlak (pasti). Hal ini didasari oleh *flowchart* (bagan alir) adalah sebuah gambaran dari hasil pemikiran dalam menganalisa suatu permasalahan dalam komputer. Karena setiap analisa akan menghasilkan hasil yang bervariasi antara satu dan lainnya. Kendati begitu secara garis besar setiap perancangan *flowchart* selalu terdiri dari tiga bagian, yaitu input, proses dan *output*.

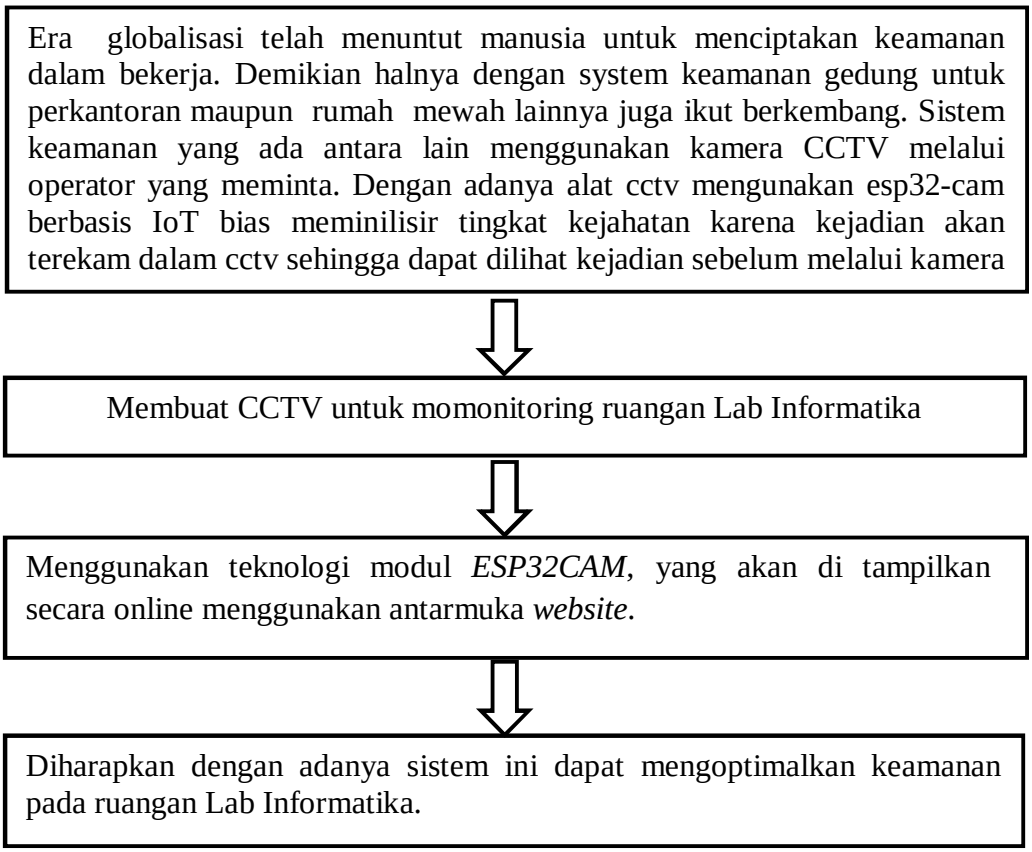
Berikut ini adalah beberapa simbol yang digunakan dalam menggambar suatu *flowchart* :

Tabel 2. 11 Simbol-simbol Flowchart

Simbol	Nama	Fungsi
	Terminator	Permulaan / akhir program
	Garis alir	Arah aliran program
	Preparation	Proses inisialisasi/pemberian harga awal
	Proses	Proses perhitungan/proses pengolahan data
	Decision	Perbandingan pernyataan penyelesaian data yang memberikan pilihan untuk laghkah selanjutnya.
	Input/Output data	Proses input/output data.
	On page	Penghubung bagian-bagian flowchart yang berada pada satu halaman

O. Kerangka Berfikir

Untuk memahami kerangka pikir penelitian ini dapat dilihat dalam bentuk diagram :



BAB III

METODE PENELITIAN

A. Lokasi dan Waktu Penelitian

1. Lokasi

   Lokasi penelitian dilakukan pada Ruangan Lab Informatika Universitas Muhammadiyah Parepare. Lokasi ini dipilih dengan pertimbangan kemudahan dalam mendapatkan data dan informasi yang akurat.

2. Waktu Penelitian

   Waktu yang dibutuhkan dalam palaksanaan penelitian ini adalah ± 3 bulan.

B. Jenis Penelitian

Jenis penelitian yang digunakan pada Rancang Bangun Memonitoring Ruangan Lab Informatika Menggunakan *ESP32-CAM* Berbasis *Internet Of Things* adalah :

1. Penelitian Deskriptif

   Penelitian Deskriptif memberikan gambaran fenomena yang sesungguhnya terjadi dalam menggambarkan fakta-fakta dan informasi secara sistematis.

2. Penelitian Kepustakaan

   Penelitian Kepustakaan adalah jenis penelitian dengan mencari informasi terkait masalah pada buku, catatan, serta berbagai laporan untuk mencocok dan mendapatkan pemecahan dari masalah yang dikaji.

C. Metode Pengumpulan Data

Untuk memperoleh data-data yang dibutuhkan dalam rangka melakukan penelitian, maka penulis mengumpulkan data melalui beberapa cara yaitu :

1. Pengumpulan Data

   Metode pengumpulan data menggunakan Metode Kepustakaan yaitu metode atau teknik pengumpulan data yang bersumber dari literatur buku-buku penunjang dan jurnal untuk konsep teori yang berhubungan dengan objek permasalahan penelitian.

2. Analisis Data

   Menganalisa data-data yang sebelumnya telah dikumpulkan dengan caramenganalisis cara kerja sistem yang akan dirancang, mengidentifikasi masalah, dan menganalisa kebutuhan sistem.

3. Perancangan Program

   Sebagai pedoman dalam penulisan program atau kode-kode agar berjalan sesuai rencana.

4. Uji Coba Program

   Pengujian program dilakukan untuk memastikan bahwa program yang dibuat dapat berjalan dengan baik dengan cara membuat *Flowchart* Sistem dan kemudian akan digunakan untuk membuat Desain Sistem.

5. Evaluasi

   Sistem yang telah selesai dibangun perlu adanya evaluasi untuk menemukan kelemahan yang terdapat pada program yang telah dibangun tadi.

D. Alat dan Bahan Penelitian

Dalam melakukan penelitian, maka diperlukan alat dan bahan penelitian yang mendukung kegiatan penelitian tersebut. Alat dan bahan yang diperlukan antara lain :

1. Alat

   Alat yang digunakan selama proses penelitian yaitu :

   a. *Laptop* HP 240 G2 Notebook PC dengan spesifikasi:

      1. *Processor* : Intel(R) CORE(TM) i3-3110M CPU @ 2.40GHz (4 CPUs) ,~2.4 GHz
      2. *RAM* : 4 GB
      3. *Hard Disk Drive* : 500 GB
      4. *Monitor* : 14.0 inci
      5. Sistem Operasi : *Windows 10 PRO 64-bit*

   b. Alat penelitian yang digunakan selama proses penelitian yaitu alat kendali berbasis elektronika dengan spesifikasi *Hardware* yaitu :

      1. *ESP32-CAM*
      2. Memori 16 GB

2. *Software* yang digunakan dalam pembuatan aplikasi yaitu :

   1. *Windows 10*
   2. *Arduino IDE*
   3. *Browser*

E. Tahapan Penelitian

Tahap-tahap penelitian yang dimaksud dalam penelitian ini adalah berkenaan dengan proses pelaksanaan penelitian.

1. Persiapan Penelitian

   Tahap persiapan adalah tahap yang dilakukan sebelum melakukan penelitian.Pada tahapan ini dimulai dengan mengkaji permasalahan yang telah ada kemudian melakukan studi literatur mengenai permasalahan yang sedang diteliti.

2. Studi Literatur

   Tahap pelaksanaan merupakan tahap penerapan penelitian. Pada tahap ini dilakukan beberapa kegiatan yang mengenai dengan penelitian yang akan dilakukan. Pada tahapan ini ada empat langkah yang harus dipenuhi guna untuk mencapai hasil maksimal dalam penelitian yaitu tahapan pengumpulan data dengan menggunakan beberapa metode pengumpulan data yang telah dijelaskan pada poin diatas, pengolahan data, analisis data dan selanjutnya penafsiran hasil analisis. Setelah kegiatan tersebut, proses selanjutnya yaitu dengan melakukan tugas lapangan dalam rangka mengumpulkan data, untuk kemudian akan diproses. Proses yang dimaksud meliputi penyuntingan, penerapan masalah dalam aplikasi program, serta analisis sebagai penarikan kesimpulan hasil akhir.

3. Pengumpulan Data

Pada tahap ini peneliti melakukan pencarian data-data di berbagai sumber untuk dikumpulkan lalu dikaji lebih lanjut.

4. Analisis

   Pada tahap analisis, peneliti melakukan analisa terhadap permasalahan yang diteliti kemudian merumuskan masalah yang menjadi pokok penelitian sehingga dapat dibuat alternatif pemecahan masalah.

5. Perancangan

   Peneliti kemudian merancang aplikasi yang ingin dibuat berdasarkan alternatif pemecahan masalah.

6. Pengujian

   Setelah melakukan perancangan, peneliti kemudian menguji hasil perancangan yang telah dibuat.Jika hasil perancangan terdapat kekurangan atau kelemahan maka kembali ke tahap analisis.

7. Implementasi

   Setelah pada perancangan tidak terdapat kekurangan maka aplikasi siap untuk di gunakan oleh *user*.

8. Tahap Penyelesaian

   Tahap penyelesaian merupakan tahap akhir yang dilakukan pada penelitian yang dilakukan.Pada tahap ini penyusunan laporan penelitian disusun.

F. Metode Pengujian

Metode pengujian yang Penulis gunakan dalam pengujian rancang bangun memonitoring ruangan lab informatika menggunakan ESP32-CAM berbasis internet of things adalah sebagai berikut :

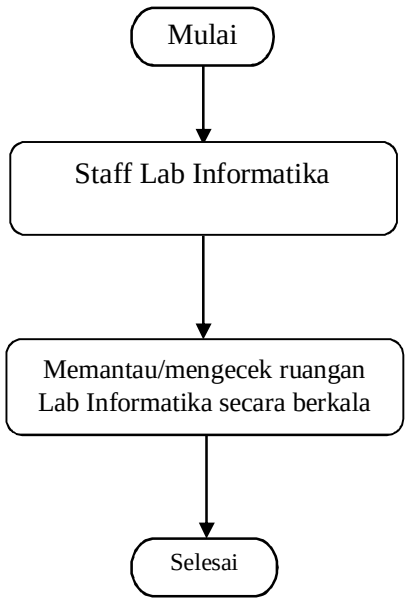
1. *Black Box*. *Black box* testing merupakan pengujian yang bertujuan untuk menunjukkan fungsi perangkat lunak tentang cara beroperasinya program, apakah pemasukan data keluaran telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya. Teknik pengujian ini berfokus pada domain informasi dari perangkat lunak, dengan melakukan *test case* dengan mempartisi domain input dari suatu program dengan cara yang memberikan cakupan pengujian yang mendalam.
2. Metode *White Box*, yaitu pengujian terhadapt cara kerja perangkat lunak itu sendiri yaitu prosedur programnya (*Basis Path*) atau proses *Looping* (Pengulangan).

BAB IV

HASIL DAN PEMBAHASAN

Pada penjelasan perancangan sistem dan analisis ini perlu diperhatikan dengan baik karena pada saat pembuatan alat dan aplikasi pertama–tama yang harus dilakukan adalah membuat rancangan *prototype* dengan jalur jalannya program. Dimulai dengan bagaimana data di *input* kemudian diproses dengan sistem yang kita buat.

A. Analisis Sistem yang Sedang Berjalan

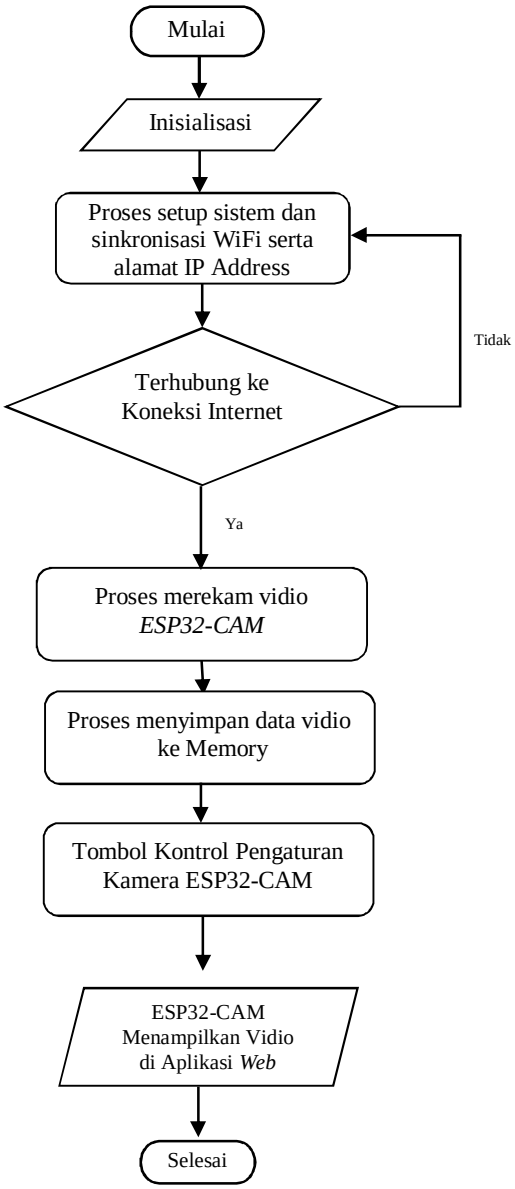


Gambar 4. 1 Sistem Yang Berjalan

Pada gambar di atas menjelaskan bagaimana sistem yang berjalan saat ini masih bersifat manual yaitu staff lab informatila masih mengecek/memantau ruangan secara berkala.

B. Rancangan Sistem yang Diusulkan

Sistem dimulai dari mengoperasikan komponen alat yang saling terhubung pada prototipe sistem *monitoring* ruangan Lab Informatika Berbasis *Internet of Things* yaitu menghubungkan *modul ESP32-CAM* dan *motor servo*.



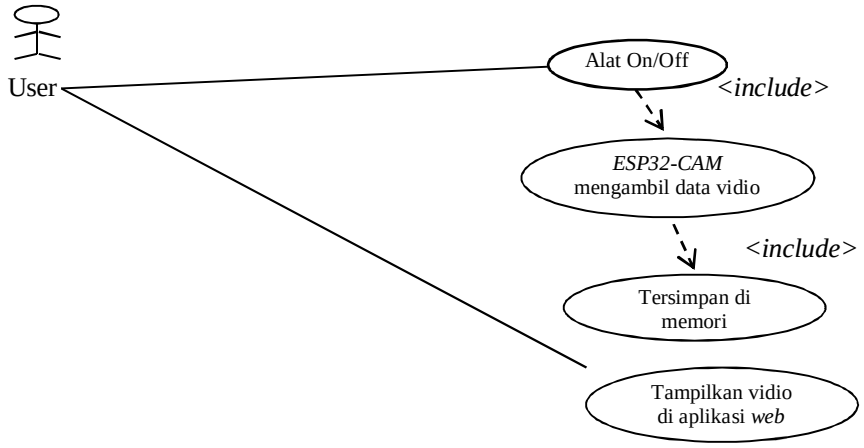
Gambar 4.2 flowchart sistem monitoring yang diusulkan

Penjelasan dari flowchart tersebut adalah pertama melakukan inisialisasi, dan sinkronisasi *Wifi* serta alamat *Ip Address* yang menjadi perantara komunikasi *web server* dengan *kontroler ESP32-CAM*. Setelah proses sinkronisasi selesai, maka sistem akan mengaktifkan *web cam* untuk melakukan proses pengambilan vidio, data Hasil vidio akan tersimpan ke memory dan dikirimkan ke aplikasi *web* untuk menampilkan vidio, Proses ini akan berlangsung terus menerus, sampai sistem dinonaktifkan.

Pemodelan dalam pengembangan sistem informasi menggunakan UML. UML suatu metode pemodelan secara visual untuk sarana perancangan sistem berorientasi objek. Tipe UML yang digunakan dalam pengembangan sistem informasi adalah *Use Case*. Diagram dibentuk sebagai berikut:

1. *Use Case*

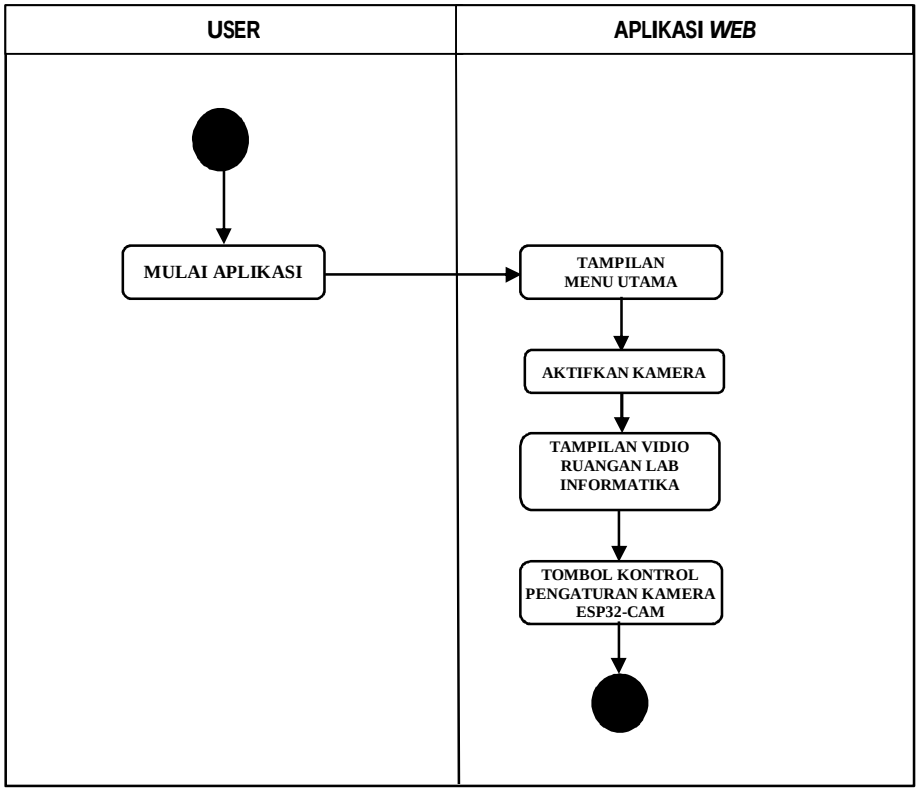
   use case dibentuk sebagai langkah awal perancangan sistem yang menggambarkan interaksi antara sistem dan aktor. *Use Case* juga dapat mendeskripsikan tipe interaksi si pemakai sistem dengan sistemnya. Berikut ini gambaran *use case* :



Gambar 4.3 *Use Case* Diagram

Pada gambar diatas merupakan perancangan sistem yang dibuat menggunakan diagram *use case*. Dalam diagram *use case* ini terdapat tiga *case*, *case* pertama adalah *ESP32-CAM* membaca data vidio, *case* yang kedua meyimpan hasil reakaman vidio ke memory, *case* yang ketiga adalah meampilkan vidio yang di aplikasi *web*.

2. Activity diagram

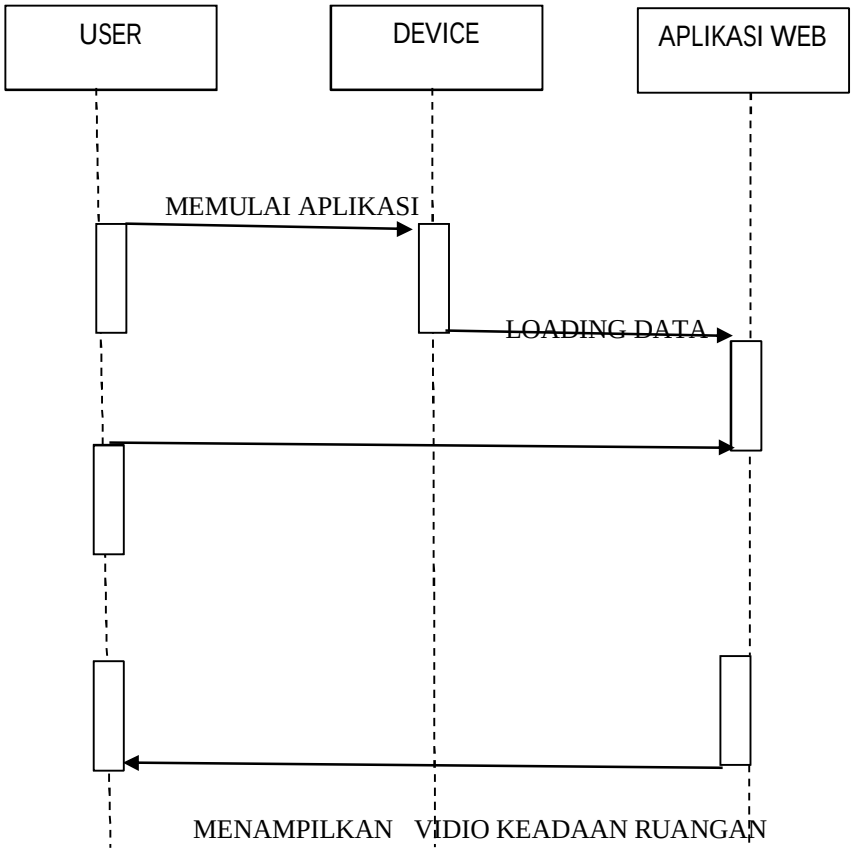


Gambar 4. 4 *Activity diagram*

Activity Diagram di atas menjelaskan bahwa ketika pengguna menjalankan aplikasi ini, pengguna akan langsung diarahkan ke tampilan aplikasi. Dimana aplikasi akan langsung menampilkan vidio keadaan ruangan Lab Informatika.

3. *Sequence* diagram

Sequence Diagram merupakan aliran antara objek yang membentuk proses, berikut adalah diagram *sequence*.



Gambar 4. 5 Sequence diagram

C. Desain Sistem

Dalam proses perancangan desain aplikasi, dalam hal ini desain tampilan perlu adanya sketsa desain yang digunakan untuk menggambarkan

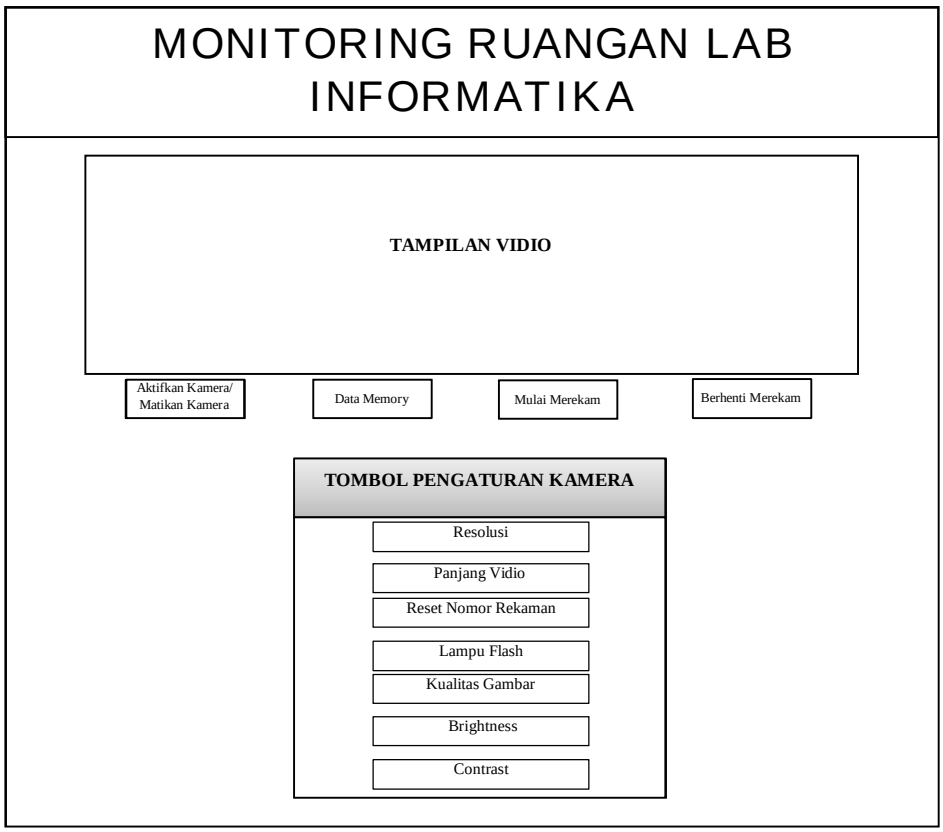
pembuatan media. Sketsa tersebut dibentuk dalam sebuah gambar. Tahap perancangan dimulai dengan membuat sketsa aplikasi agar pembuatan program aplikasi dapat dilakukan secara terurut.

1. Perancangan Antar Muka / Interface

Berikut penjelasan dari beberapa rancangan antar muka dari aplikasi ini:

a) Halaman Utama

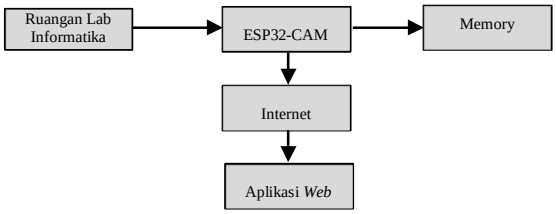
Halaman utama ini merupakan halaman yang berisi tampilan vidio untuk memonitoring ruangan Lab Informatika, dan pada tombol kontrol berfungsi untuk menggerkkan *ESP32-CAM*.



Gambar 4. 6 desain perancangan antarmuka aplikasi web

2. Blok Diagram Sistem

Terdapat dua bagian pengerjaan pada *system monitoring* ruangan lab informatika yang meliputi *hardware* dan *software*. Dimana pada bagian *hardware* meliputi pengerjaan pembuatan alat *system monitoring*. Lalu pada bagian *software* meliputi pengerjaan pembuatan aplikasi.

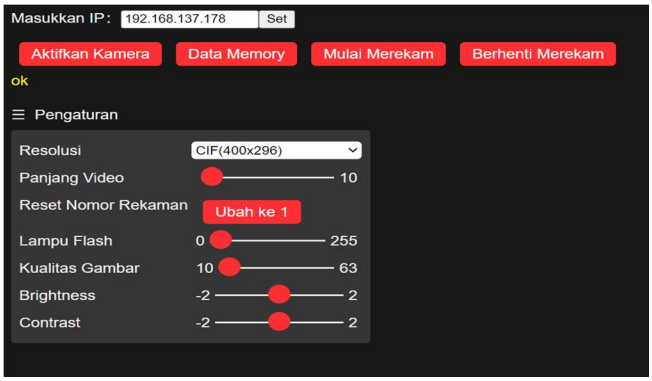


Gambar 4.7 *Blok diagram system*

D. Rancangan Sistem

Berikut adalah rancangan dari sistem rancang bangun memonitoring ruangan lab informatika menggunakan *ESP32-CAM* berbasis *internet of things*

1. Rancangan *Software*

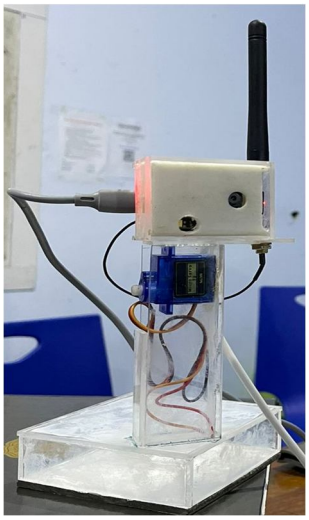


Gambar 4.8 TampilanMenu Utama

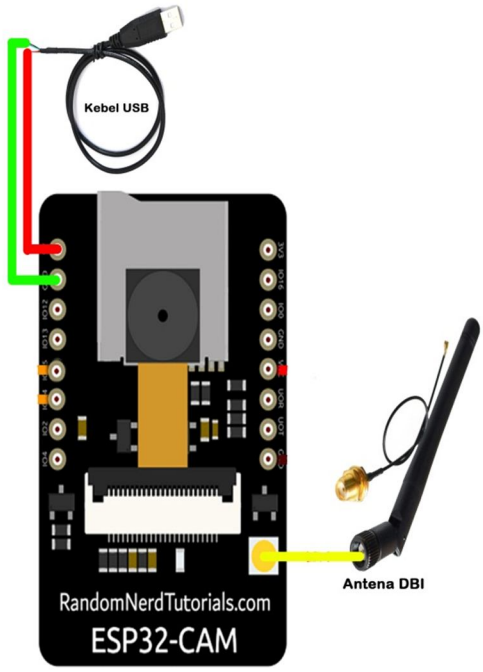
Software sistem memonitoring ruangan lab informatika menggunakan *ESP32-CAM* dirancang untuk memantau keadaan ruangan secara *Realtime* yang dikontrol di ruangan tertentu. Sistem memonitoring ruangan akan diaktifkan melalui *Software* ini kemudian data vidio pun akan di tampilkan pada apalikasi *web*, dan terdapa juga tombol kontrol pengaturan kamera.

2. Rancangan *Hardware*

Rancangan *Hardware* dirangkai dalam bentuk Miniatur sederhana seperti yang terlihat pada gambar dibawah.



Gambar 4. 9 bentuk konstruksi rangkaian *hardware*



Gambar 4.10 Rancangan Rangkaian *Hardware*

Pada perancangan perangkat keras meliputi penjelasan komponen-komponen yang digunakan. Adapun uraian penghubungan pada setiap alat adalah sebagai berikut :

a. Koneksi *Pin* kabel USB

Tabel 4. 1 Koneksi kabel USB

NO	***PIN***	**Keterangan**
1	Kabel merah	Dihubungnkan ke pin 5v *ESP32-CAM*
2	Kabel merah	Dihubungkan ke pin GND *ESP32-CAM*

E. Rancangan Perangkat Lunak (*Software*)

Setelah melakukan rancangan perangkat keras selanjutnya adalah melakukan perancangan perangkat lunak. Pada bagian perangkat lunak ini merupakan bagian pembuatan program yang akan di *upload* ke dalam *mikrokontroler* yaitu berupa perintah untuk mengelola data untuk diaplikasikan. Adapun bahasa pemrograman yang digunakan adalah berupa *sketch* program dalam bahasa C dan untuk membangun sistem rancang bangun memonitoring ruangan lab menggunakan pemrograman bahasa Html (*hypertext markup language)*.



CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
bool configfile = false;
bool InternetOff = true;
bool reboot_now = false;
String cssid;
String cpass;
String czone;

TaskHandle_t the_camera_loop_task;
TaskHandle_t the_sd_loop_task;
TaskHandle_t the_streaming_loop_task;

SemaphoreHandle_t wait_for_sd;
SemaphoreHandle_t sd_go;

long current_frame_time;
long last_frame_time;

#define fbs 8 // was 64 -- how many kb of static ram for psram -> sram buffer for sd write
uint8_t framebuffer_static[fbs * 1024 + 20];

typedef struct {
        size_t size; //number of values used for filtering
        size_t index; //current value index
        size_t count; //value count
        int sum;
        int * values; //array to be filled with values
} ra_filter_t;

typedef struct {
        httpd_req_t *req;
        size_t len;
} jpg_chunking_t;

#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\n\r\n";

static ra_filter_t ra_filter;
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
static ra_filter_t ra_filter;
httpd_handle_t stream_httpd = NULL;
httpd_handle_t camera_httpd = NULL;

static ra_filter_t * ra_filter_init(ra_filter_t * filter, size_t sample_size){
    memset(filter, 0, sizeof(ra_filter_t));

    filter->values = (int *)malloc(sample_size * sizeof(int));
    if(!filter->values){
        return NULL;
    }
    memset(filter->values, 0, sample_size * sizeof(int));

    filter->size = sample_size;
    return filter;
}

static int ra_filter_run(ra_filter_t * filter, int value){
    if(!filter->values){
        return value;
    }
    filter->sum -= filter->values[filter->index];
    filter->values[filter->index] = value;
    filter->sum += filter->values[filter->index];
    filter->index++;
    filter->index = filter->index % filter->size;
    if (filter->count < filter->size) {
        filter->count++;
    }
    return filter->sum / filter->count;
}

// CAMERA_MODEL_AI_THINKER
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
115 #define Y8_GPIO_NUM       34
116 #define Y7_GPIO_NUM       39
117 #define Y6_GPIO_NUM       36
118 #define Y5_GPIO_NUM       21
119 #define Y4_GPIO_NUM       19
120 #define Y3_GPIO_NUM       18
121 #define Y2_GPIO_NUM        5
122 #define VSYNC_GPIO_NUM    25
123 #define HREF_GPIO_NUM     23
124 #define PCLK_GPIO_NUM     22
125
126 camera_fb_t * fb_curr = NULL;
127 camera_fb_t * fb_next = NULL;
128
129 #include <stdio.h>
130 #include "freertos/FreeRTOS.h"
131 #include "freertos/task.h"
132 #include "esp_system.h"
133 #include "nvs_flash.h"
134 #include "nvs.h"
135 #include "soc/soc.h"
136 #include "soc/cpu.h"
137 #include "soc/rtc_cntl_reg.h"
138
139 static esp_err_t cam_err;
140 float most_recent_fps = 0;
141 int most_recent_avg_framesize = 0;
142
143 uint8_t* framebuffer;
144 int framebuffer_len;
145
146 int first = 1;
147 long frame_start = 0;
148 long frame_end = 0;
149 long frame_total = 0;
150 long frame_average = 0;
151 long loop_average = 0;
152 long loop_total = 0;
153 long total_frame_data = 0;
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
154 long last_frame_length = 0;
155 int done = 0;
156 long avi_start_time = 0;
157 long avi_end_time = 0;
158 int start_record = 0;
159
160 int we_are_already_stopped = 0;
161 long total_delay = 0;
162 long bytes_before_last_100_frames = 0;
163 long time_before_last_100_frames = 0;
164
165 long time_in_loop = 0;
166 long time_in_camera = 0;
167 long time_in_sd = 0;
168 long time_in_good = 0;
169 long time_total = 0;
170 long time_in_web1 = 0;
171 long time_in_web2 = 0;
172 long delay_wait_for_sd = 0;
173 long wait_for_cam = 0;
174
175 int do_it_now = 0;
176
177 // MicroSD
178 #include "driver/sdmmc_host.h"
179 #include "driver/sdmmc_defs.h"
180 #include "sdmmc_cmd.h"
181 #include "esp_vfs_fat.h"
182 #include "FS.h"
183 #include <SD_MMC.h>
184
185 File logfile;
186 File avifile;
187 File idxfile;
188
189 char avi_file_name[100];
190
191 static int i = 0;
192 uint16_t frame_cnt = 0;
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
static int i = 0;
uint16_t frame_cnt = 0;
uint16_t remnant = 0;
uint32_t length = 0;
uint32_t startms;
uint32_t elapsedms;
uint32_t uVideoLen = 0;

int bad_jpg = 0;
int extend_jpg = 0;
int normal_jpg = 0;

int file_number = 0;
int file_group = 0;
long boot_time = 0;

long totalp;
long totalw;

#define BUFFSIZE 512

uint8_t buf[BUFFSIZE];

#define AVIOFFSET 240 // AVI main header length

unsigned long movi_size = 0;
unsigned long jpeg_size = 0;
unsigned long idx_offset = 0;

uint8_t zero_buf[4] = {0x00, 0x00, 0x00, 0x00};
uint8_t dc_buf[4] = {0x30, 0x30, 0x64, 0x63};    // "00dc"
uint8_t dc_and_zero_buf[8] = {0x30, 0x30, 0x64, 0x63, 0x00, 0x00, 0x00, 0x00};

uint8_t avi1_buf[4] = {0x41, 0x56, 0x49, 0x31};    // "AVI1"
uint8_t idx1_buf[4] = {0x69, 0x64, 0x78, 0x31};    // "idx1"

struct frameSizeStruct {
  uint8_t frameWidth[2];
  uint8_t frameHeight[2];
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
static const frameSizeStruct frameSizeData[] = {
  {{0x60, 0x00}, {0x60, 0x00}}, // FRAMESIZE_96X96,    // 96x96
  {{0xA0, 0x00}, {0x78, 0x00}}, // FRAMESIZE_QQVGA,    // 160x120
  {{0xB0, 0x00}, {0x90, 0x00}}, // FRAMESIZE_QCIF,     // 176x144
  {{0xF0, 0x00}, {0xB0, 0x00}}, // FRAMESIZE_HQVGA,    // 240x176
  {{0xF0, 0x00}, {0xF0, 0x00}}, // FRAMESIZE_240X240,  // 240x240
  {{0x40, 0x01}, {0xF0, 0x00}}, // FRAMESIZE_QVGA,     // 320x240   framessize
  {{0x90, 0x01}, {0x28, 0x01}}, // FRAMESIZE_CIF,      // 400x296       bytes per buffer required in psram - quality must be higher number (lower quality) than config quality
  {{0xE0, 0x01}, {0x40, 0x01}}, // FRAMESIZE_HVGA,     // 480x320       low qual  med qual  high quality
  {{0x80, 0x02}, {0xE0, 0x01}}, // FRAMESIZE_VGA,      // 640x480   8   11+   ##  6-10  ##  0-5         indoor(56,COUNT=3)  (56,COUNT=2)          (56,count=1)
                                                       //               38,400    61,440    153,600
  {{0x20, 0x03}, {0x58, 0x02}}, // FRAMESIZE_SVGA,     // 800x600   9
  {{0x00, 0x04}, {0x00, 0x03}}, // FRAMESIZE_XGA,      // 1024x768  10
  {{0x00, 0x05}, {0xD0, 0x02}}, // FRAMESIZE_HD,       // 1280x720  11  115,200   184,320   460,800     (11)50.000  25.4fps   (11)50.000 12fps    (11)50,000  12.7fps
  {{0x00, 0x05}, {0x00, 0x04}}, // FRAMESIZE_SXGA,     // 1280x1024 12
  {{0x40, 0x06}, {0xB0, 0x04}}, // FRAMESIZE_UXGA,     // 1600x1200 13  240,000   384,000   960,000
  // 3MP Sensors
  {{0x80, 0x07}, {0x38, 0x04}}, // FRAMESIZE_FHD,      // 1920x1080 14  259,200   414,720   1,036,800   (11)210,000 5.91fps
  {{0xD0, 0x02}, {0x00, 0x05}}, // FRAMESIZE_P_HD,     //  720x1280 15
  {{0x60, 0x03}, {0x00, 0x06}}, // FRAMESIZE_P_3MP,    //  864x1536 16
  {{0x00, 0x08}, {0x00, 0x06}}, // FRAMESIZE_QXGA,     // 2048x1536 17  393,216   629,146   1,572,864
  // 5MP Sensors
  {{0x00, 0x0A}, {0xA0, 0x05}}, // FRAMESIZE_QHD,      // 2560x1440 18  460,800   737,280   1,843,200   (11)400,000 3.5fps    (11)330,000 1.95fps
  {{0x00, 0x0A}, {0x40, 0x06}}, // FRAMESIZE_WQXGA,    // 2560x1600 19
  {{0x38, 0x04}, {0x80, 0x07}}, // FRAMESIZE_P_FHD,    // 1080x1920 20
  {{0x00, 0x0A}, {0x80, 0x07}}  // FRAMESIZE_QSXGA,    // 2560x1920 21  614,400   983,040   2,457,600   (15)425,000 3.25fps   (15)382,000 1.7fps  (15)385,000 1.7fps

};

const int avi_header[AVIOFFSET] PROGMEM = {
  0x52, 0x49, 0x46, 0x46, 0xD8, 0x01, 0x0E, 0x00, 0x41, 0x56, 0x49, 0x20, 0x4C, 0x49, 0x53, 0x54,
  0xD0, 0x00, 0x00, 0x00, 0x68, 0x64, 0x72, 0x6C, 0x61, 0x76, 0x69, 0x68, 0x38, 0x00, 0x00, 0x00,
  0xA0, 0x86, 0x01, 0x00, 0x80, 0x66, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00,
  0x64, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x80, 0x02, 0x00, 0x00, 0xe0, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x4C, 0x49, 0x53, 0x54, 0x84, 0x00, 0x00, 0x00,
  0x73, 0x74, 0x72, 0x6C, 0x73, 0x74, 0x72, 0x68, 0x30, 0x00, 0x00, 0x00, 0x76, 0x69, 0x64, 0x73,
  0x4D, 0x4A, 0x50, 0x47, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x01, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

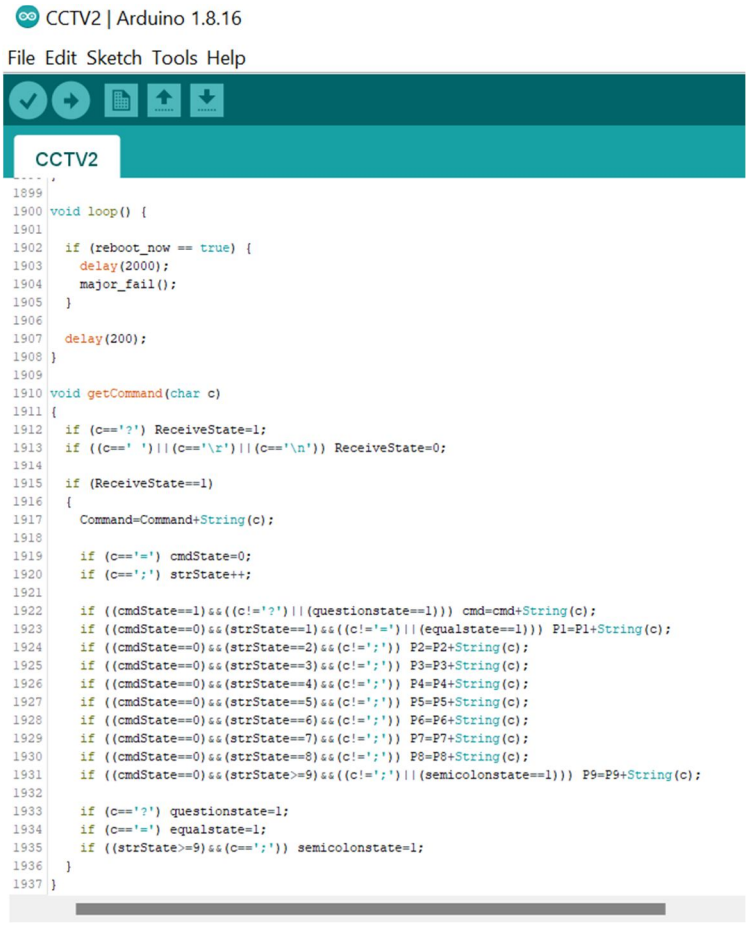
```
284
285   uint8_t y[4];
286   y[0] = i % 0x100;
287   y[1] = (i >> 8) % 0x100;
288   y[2] = (i >> 16) % 0x100;
289   y[3] = (i >> 24) % 0x100;
290   size_t il_err = fd.write(y , 4);
291 }
292
293 static void inline print_2quartet(unsigned long i, unsigned long j, File fd) {
294
295   uint8_t y[8];
296   y[0] = i % 0x100;
297   y[1] = (i >> 8) % 0x100;
298   y[2] = (i >> 16) % 0x100;
299   y[3] = (i >> 24) % 0x100;
300   y[4] = j % 0x100;
301   y[5] = (j >> 8) % 0x100;
302   y[6] = (j >> 16) % 0x100;
303   y[7] = (j >> 24) % 0x100;
304   size_t il_err = fd.write(y , 8);
305 }
306
307 void major_fail() {
308
309   Serial.println(" ");
310   logfile.close();
311
312   for  (int i = 0;  i < 3; i++) {
313     for (int j = 0; j < 3; j++) {
314       digitalWrite(33, LOW);   delay(150);
315       digitalWrite(33, HIGH);  delay(150);
316     }
317     delay(1000);
318
319     for (int j = 0; j < 3; j++) {
320       digitalWrite(33, LOW);  delay(500);
321       digitalWrite(33, HIGH); delay(500);
322     }
```

CCTV2 | Arduino 1.8.16

File Edit Sketch Tools Help

CCTV2

```
230 };
231
232 void the_camera_loop (void* pvParameter);
233 void the_sd_loop (void* pvParameter);
234 void delete_old_stuff();
235
236 static const framesizeStruct framesizeData[] = {
237   {{0x60, 0x00}, {0x60, 0x00}}, // FRAMESIZE_96X96,    // 96x96
238   {{0xA0, 0x00}, {0x78, 0x00}}, // FRAMESIZE_QQVGA,    // 160x120
239   {{0xB0, 0x00}, {0x90, 0x00}}, // FRAMESIZE_QCIF,     // 176x144
240   {{0xF0, 0x00}, {0xB0, 0x00}}, // FRAMESIZE_HQVGA,    // 240x176
241   {{0xF0, 0x00}, {0xF0, 0x00}}, // FRAMESIZE_240X240,  // 240x240
242   {{0x40, 0x01}, {0xF0, 0x00}}, // FRAMESIZE_QVGA,     // 320x240   framessize
243   {{0x90, 0x01}, {0x28, 0x01}}, // FRAMESIZE_CIF,      // 400x296       bytes per buffer required in psram - quality must be higher number (lower quality) than config quality
244   {{0xE0, 0x01}, {0x40, 0x01}}, // FRAMESIZE_HVGA,     // 480x320       low qual  med qual  high quality
245   {{0x80, 0x02}, {0xE0, 0x01}}, // FRAMESIZE_VGA,      // 640x480   8   11+   ##  6-10  ##  0-5        indoor(56,COUNT=3)  (56,COUNT=2)          (56,count=1)
246                                 //                     38,400    61,440    153,600
247   {{0x20, 0x03}, {0x58, 0x02}}, // FRAMESIZE_SVGA,     // 800x600   9
248   {{0x00, 0x04}, {0x00, 0x03}}, // FRAMESIZE_XGA,      // 1024x768  10
249   {{0x00, 0x05}, {0xD0, 0x02}}, // FRAMESIZE_HD,       // 1280x720  11  115,200   184,320   460,800     (11)50.000  25.4fps   (11)50.000 12fps    (11)50,000  12.7fps
250   {{0x00, 0x05}, {0x00, 0x04}}, // FRAMESIZE_SXGA,     // 1280x1024 12
251   {{0x40, 0x06}, {0xB0, 0x04}}, // FRAMESIZE_UXGA,     // 1600x1200 13  240,000   384,000   960,000
252   // 3MP Sensors
253   {{0x80, 0x07}, {0x38, 0x04}}, // FRAMESIZE_FHD,      // 1920x1080 14  259,200   414,720   1,036,800   (11)210,000 5.91fps
254   {{0xD0, 0x02}, {0x00, 0x05}}, // FRAMESIZE_P_HD,     //  720x1280 15
255   {{0x60, 0x03}, {0x00, 0x06}}, // FRAMESIZE_P_3MP,    //  864x1536 16
256   {{0x00, 0x08}, {0x00, 0x06}}, // FRAMESIZE_QXGA,     // 2048x1536 17  393,216   629,146   1,572,864
257   // 5MP Sensors
258   {{0x00, 0x0A}, {0xA0, 0x05}}, // FRAMESIZE_QHD,      // 2560x1440 18  460,800   737,280   1,843,200   (11)400,000 3.5fps    (11)330,000 1.95fps
259   {{0x00, 0x0A}, {0x40, 0x06}}, // FRAMESIZE_WQXGA,    // 2560x1600 19
260   {{0x38, 0x04}, {0x80, 0x07}}, // FRAMESIZE_P_FHD,    // 1080x1920 20
261   {{0x00, 0x0A}, {0x80, 0x07}}  // FRAMESIZE_QSXGA,    // 2560x1920 21  614,400   983,040   2,457,600   (15)425,000 3.25fps   (15)382,000 1.7fps  (15)385,000 1.7fps
262
263 };
264
265 const int avi_header[AVIOFFSET] PROGMEM = {
266   0x52, 0x49, 0x46, 0x46, 0xD8, 0x01, 0x0E, 0x00, 0x41, 0x56, 0x49, 0x20, 0x4C, 0x49, 0x53, 0x54,
267   0xD0, 0x00, 0x00, 0x00, 0x68, 0x64, 0x72, 0x6C, 0x61, 0x76, 0x69, 0x68, 0x38, 0x00, 0x00, 0x00,
268   0xA0, 0x86, 0x01, 0x00, 0x80, 0x66, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00,
```



Gambar 4. 11 Rancangan Perangkat Lunak

Pada gambar *arduino ide* (*sketch*) sebuah *software* untuk memprogram *arduino* dan komponen pendukung lainnya dengan menggunakan *sintaks* pemrograman bahasa C. *Arduino ide* dibuat dari bahasa pemrograman Java yang dilengkapi dengan *library* C/C++ yang membuat operasi *input/output* lebih mudah.

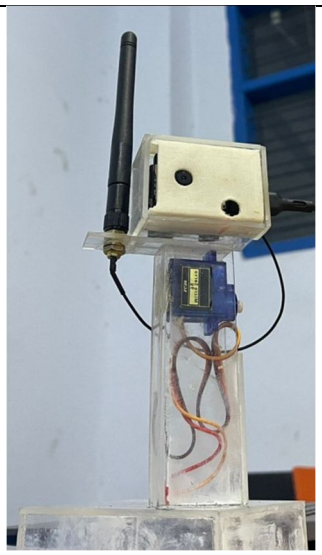
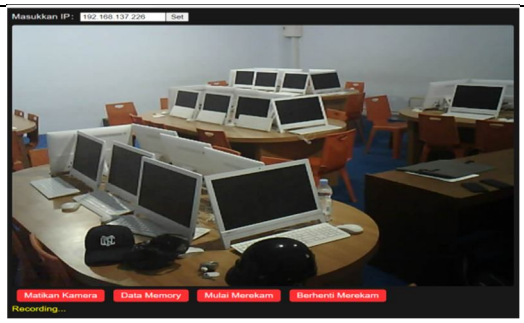
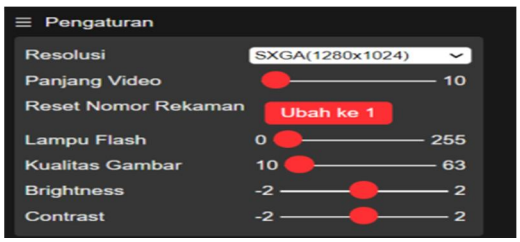
F. Pengujian Sistem

Teknik pengujian sistem merupakan proses pengeksekusian program secara keseluruhan. Pengujian sistem ini berfokus untuk mengetahui proses dan cara kerja aplikasi. Adapun teknik pengujian yang digunakan dalam rancang bangun memonitoring ruangan lab informatika menggunakan *ESP32-CAM* Berbasis *Internet Of Things* Pengujian Alat secara langsung, Pengujian *Black Box* dan Pengujian *White Box*.

1. Pengujian *Black Box*

a. Pengujian tahap pertama

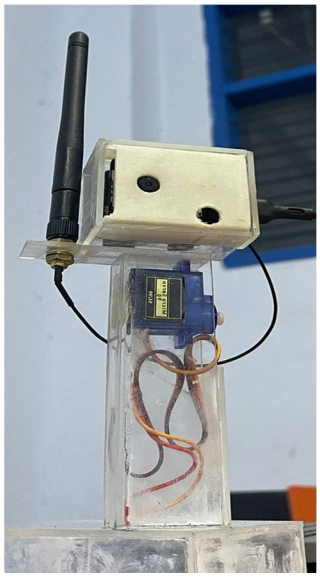
Tabel 4.4 Pengujian tahap pertama

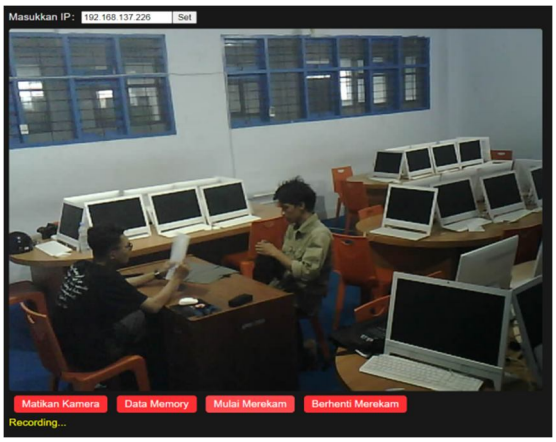
Uji coba	Hasil	Keterangan
Proses merekam vidio dari sisi depan kiri ruangan dengan menyimpan hasil rekaman setiap Durasi 10 detik	√	Berhasil, karena setiap hasil rekaman mencapai 10 detik maka vidio akan tersimpan di data memory secara otomatis.
Foto dan *Screenshot*		
  		

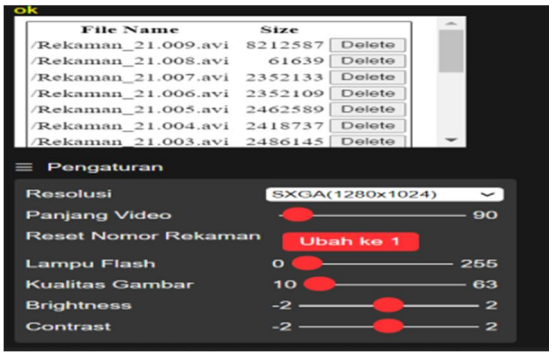
b. Pengujian tahap kedua

Tabel 4.5 Pengujian tahap kedua

Uji coba	Hasil	Keterangan
Proses merekam vidio dari sisi depan kanan ruangan dengan menyimpan hasil rekaman setiap Durasi 90 detik	√	Berhasil, karena setiap hasil rekaman mencapai 90 detik maka vidio akan tersimpan di data memory secara otomatis.
Foto dan *Screenshot*		

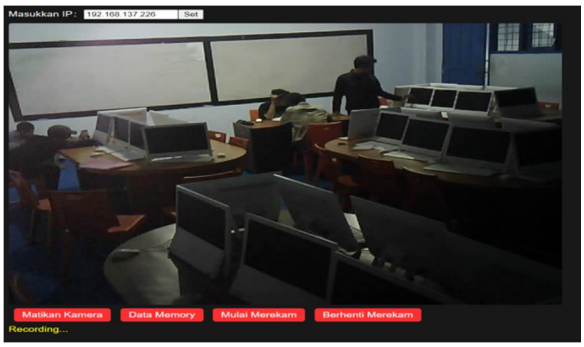
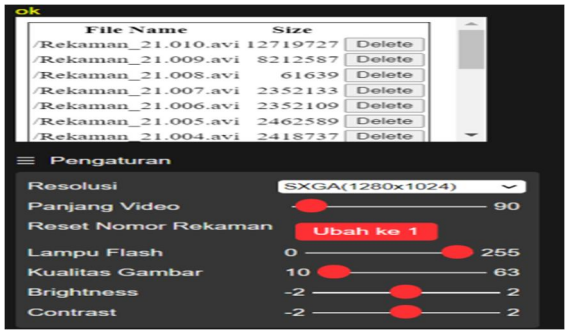






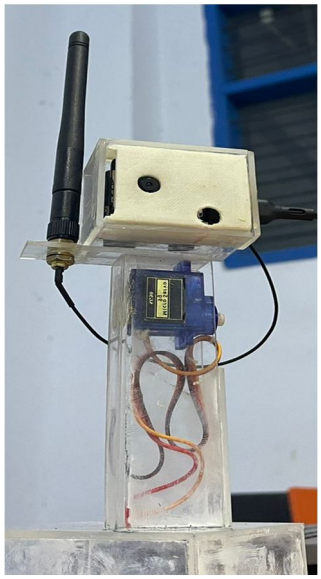
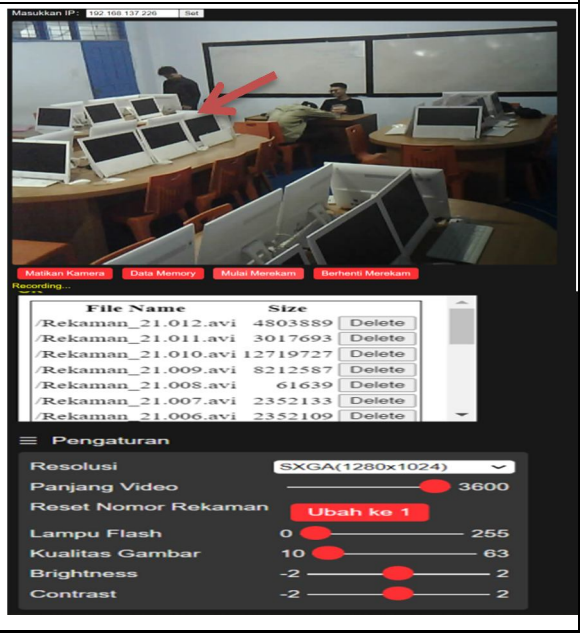
c. Pengujian tahap ketiga

Tabel 4.6 Pengujian tahap ketiga

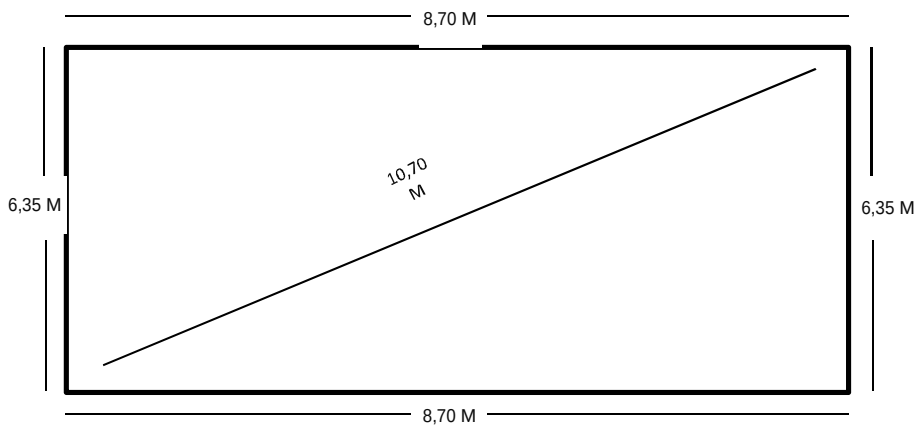
Uji coba	Hasil	Keterangan
Proses merekam vidio dari sisi belakang kiri ruangan dengan menyimpan hasil rekaman setiap durasi 90 detik dengan mengaftikan Lampu Flash.	√	Berhasil, karena setiap hasil rekaman mencapai 90 detik maka vidio akan tersimpan di data memory secara otomatis dan lampu flash menyala.
Foto dan *Screenshot*		
  		

d. Pengujian tahap keempat

Tabel 4.6 Pengujian tahap kedua

Uji coba	Hasil	Keterangan
Proses merekam vidio dari sisi belakang kanan ruangan dengan menyimpan hasil rekaman setiap durasi 3600 detik dengan jarak kamera dengan gambar 8 meter	√	Berhasil, karena setiap hasil rekaman mencapai 3600 detik maka vidio akan tersimpan di data memory secara otomatis.
Foto dan *Screenshot*		
 		

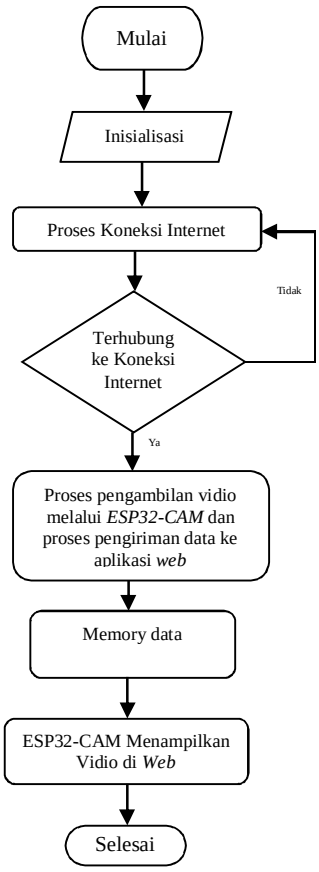
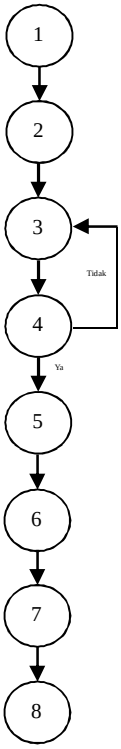
e. Luas Rungan Lab informatika Universitas muhammadiyah parepare



f. Pengujian tahap keempat

2. Pengujian *White Box*

a. Tabel 4.8 *Flowchart* dan *Flowgrap*

Flowchart	*Flowgraph*
	

Dari gambar *Flowgraph* di atas dapat dilakukan proses perhitungan sebagai berikut:

1) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

DenganRumus : V (G) = E – N + 2

Dengan E (*edge*) = 8

N (*Node*) = 8

P (Predikat *Node*) = 1

Penyelesaian : V(G) = E – N + 2

= 8 – 8 + 2

= 2

Predikat *Node* (P) = P + 1

= 1 + 1

= 2

2) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

3) *Independent Path* pada *Flowgraph* diatas adalah :

Path 1 = 1 – 2 – 4 – 3

Path 2 = 1 – 2 – 3 – 4 – 5– 6 – 7 – 8

4) *Graph Matrix*

5)

	1	2	3	4	5	6	7	8	Koneksi
1		1							1-1= 0
2			1						1-1= 0
3				1					1-1= 0
4			1		1				2-1= 1
5						1			1-1= 0
6							1		1-1= 0
7								1	1-1= 0
8									

Cyclomatic Complexity = 1+1 =2

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan hasil penelitian dan pembahasan yang telah dijelaskan pada bab- bab sebelumnya, maka dapat ditarik kesimpulan sebagai berikut:

1. Mengontrol ruangan lab informatika menggunakan pengontrol *ESP32-CAM* sebagai penghubung ke *wifi*, memory sebagai tempat penyimpanan hasil rekaman vidio sesuai durasi panjang vidio yang diinginkan
2. kamera padaa *ESP32-CAM* bekerja dengan baik sebagai kamera pengawas dengan menampilkan kondisi di sekitar ruangan.

A. Saran

Adapun saran yang penulis berikan sebagai berikut:

1. Sebaiknya menambahkan jenis sensor yang lain dan juga menambah rancangan sistem lainnya agar pengaplikasian alat ini dapat lebih dikembangkan.

DAFTAR PUSTAKA

Ahmad Shukri Mohd.2018:5 *"Pengertian Perancangan"*, Penerbit buku UTM, Yogyakarta.

Burange, A. W., & Misalkar, H. D. (2019). *"Review of Internet of Things in Development of Smart Cities with Data Management & Privacy"*.

Dwi Ely Kurniawan, Syafaruddin. 2019. *Perancangan system kamera pengawas berbasis perangkat bergerak menggunakan easpberry pi. Politeknik Negeri Batam. Jurnal Ilmiah Teknologi INformasi Terapan*. 2(15)

Ebraheim Alsaadi, Abdallah Tubaishat. (2018). *Internet of Things: Features, Challenges.*

Hemlata Channel, Sukhesh Kothari, Dipali Kadam. (2018). *Multidisciplinary Model for Smart Agriculture using Internet-of-Things (IoT), Sensors, Cloud-Computing, Mobile-Computing & Big-Data Analysis.* Department of CE, PICT, Pune, India.

Jogiyanto & Magistro : 662-663,2018"Bagian alir *Flowchart*" menurut ANSI (*American National Standard Institute*).

Kadir Abdul.2015 "Sasaran Sistem" Penerbit buku Andi, Yogyakarta.

Kevin Aston dan Peter Waher.2018 " *Internet of things (IOT)"*

Literature Review. India : *National Institute of Industrial Engineering* (NITIE).

M.U. Farooq, Sadia Mazhar, Anjum Khairi, Talha Kamal. (2018). A *Review on the Internet of Things (IoT).*

MADCOM. 2016. Pemrograman PHP dan MySQL Untuk Pemula. Yogyakarta: C.V Andi.

Nurul Hidayah, *"Konsep Dasar Sistem"* 22 Agustus 2016

NS, A. P. (2018). *Analisis dan Perancangan Sistem Informasi Manajemen Keuagan Daerah: Notasi Pemodelan Unifed Modelling Language(UML).* Bandung: Abdi Sistematika.

Okkita Rizan, Hamidah. (2019) Rancangan aplikas monitoring kamera cctv untuk perangkat mobile berbasis android.STIMK Luhur Pangkal pinang.

Raharjo, Budi. 2018. *Mudah Belajar C#(Pemograman C# dan Visual C#).* Bandung: Informatika.

Somayya Madakam, R. Ramaswamy, Siddharth Tripathi. (2018). *Internet of Things (IoT): A*

Setiawan D, Jhoni E. dan Cosmas E.S. 2019. Perancangana system pengontrol keamanan rumah dengan *smart cctv* mengunakan *arduino* berbasis *telegram,* Universitas Putera batam.InfoTekJar. 4(1).

Shubham Bhatia, Abhishek Chauhan, Vaibhav K. Nigam. (2018). *The Internet of Things: A Survey on Technology and Trends.*

Yasha. (3 Agustus 2018), *"Internet of Things"*, Panduan Lengkap. Diperoleh dari: https://www.*dewaweb*.com/blog/internet-of-things/

Vulnerabilities. International Journal of Advanced *Computer Science* and *Information Technology* (IJACSIT). Zayed University.