BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Sistem Kontrol Otomatis

Sistem kendali otomatis merupakan teknologi yang memungkinkan pengendalian proses atau perangkat tanpa intervensi langsung dari manusia. Menurut Kurniawan et al. (2025:2), dasar-dasar sistem kendali otomatis melibatkan tiga komponen utama yang saling terintegrasi, yaitu sensor untuk mengumpulkan data, pengontrol yang memproses data tersebut dan membuat keputusan, serta aktuator yang melakukan tindakan yang diperlukan berdasarkan keputusan tersebut. Komponen-komponen ini membentuk sebuah sistem yang mampu beroperasi secara mandiri, sehingga dapat mengoptimalkan proses operasi, meminimalkan kesalahan manusia, dan meningkatkan produktivitas dalam berbagai sektor industri maupun kehidupan sehari-hari.

Dalam pengertian yang lebih teknis, Syaputra et al. (2025:91) menyatakan bahwa sistem kontrol otomatis dapat didefinisikan sebagai seperangkat alat mekanik atau elektronik yang mengatur perangkat atau sistem lain dengan cara *loop kontrol*. Sistem ini umumnya terkomputerisasi dan berjalan secara otomatis, yang memungkinkan pengendalian yang lebih presisi dan konsisten. Penggunaan mekanisme *loop kontrol* memastikan bahwa sistem dapat memantau output secara berkelanjutan dan melakukan

penyesuaian secara otomatis untuk mencapai kondisi yang diinginkan, sehingga meningkatkan efisiensi dan keandalan operasional.

Perkembangan teknologi *Internet of Things* (IoT) telah membawa evolusi signifikan dalam implementasi sistem kontrol otomatis. Syaputra et al. (2025:109) menjelaskan bahwa perangkat keras dalam sistem kontrol otomatis berbasis IoT merupakan sekumpulan komponen fisik yang dirancang membentuk suatu sistem pengendali otomatis yang terhubung ke jaringan internet. Konektivitas ini memungkinkan proses pemantauan dan pengendalian dilakukan secara *real-time* dari jarak jauh, sehingga memberikan fleksibilitas operasional yang lebih tinggi. Integrasi dengan jaringan internet juga membuka peluang untuk pengumpulan data dalam skala besar, analisis prediktif, serta pengambilan keputusan yang lebih cerdas berdasarkan informasi yang diperoleh dari berbagai sumber sensor.

Mekanisme kerja sistem kontrol otomatis dapat dijelaskan melalui proses pengukuran, pengolahan, dan eksekusi yang berlangsung secara berkelanjutan. Rahman et al. (2024:646) menyatakan bahwa sistem kontrol otomatis adalah suatu sistem yang secara otomatis mengendalikan proses atau perangkat tanpa intervensi langsung manusia. Dalam operasionalnya, sensor digunakan untuk mengukur variabel lingkungan atau kondisi proses, data tersebut diolah oleh kontroler untuk mengambil keputusan, dan kontroler memberikan perintah kepada aktuator untuk melakukan penyesuaian terhadap kondisi sistem. Dengan demikian, implementasi sistem kendali otomatis akan sangat bergantung pada kerja sama antara ketiga komponen

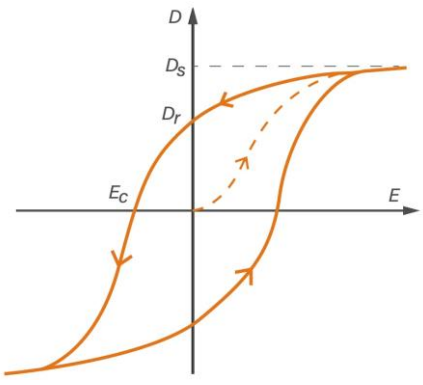
tersebut untuk menciptakan sistem yang responsif, efisien, dan mampu beradaptasi dengan perubahan kondisi lingkungan secara mandiri.

2. Algoritma *Hysteresis*

Algoritma histeresis merupakan metode kontrol yang bekerja berdasarkan prinsip *threshold* ganda, di mana sistem akan mengaktifkan aktuator ketika variabel yang dikontrol berada di luar rentang setpoint yang telah ditentukan, dan menonaktifkannya ketika nilai tersebut telah memenuhi batas yang diinginkan. Thobroni et al. (2024:278) dalam penelitiannya tentang sistem kontrol suhu dan kelembapan untuk pembuatan pupuk kompos menjelaskan bahwa cara kerja metode histerisis adalah ketika nilai suhu dan kelembapan tidak mencapai setpoint yang telah ditentukan maka otomatis aktuator yang telah terhubung dengan mikrokontroler akan menyala, dan jika nilai sudah terpenuhi aktuator akan otomatis berhenti. Mekanisme ini memungkinkan sistem untuk beroperasi secara otomatis tanpa intervensi manual, sehingga sangat efektif untuk aplikasi yang memerlukan kontrol kontinu terhadap parameter lingkungan. Keunggulan utama dari pendekatan ini terletak pada kemampuannya untuk mengurangi frekuensi switching aktuator yang berlebihan, sehingga tidak hanya meningkatkan efisiensi energi tetapi juga memperpanjang umur pakai komponen mekanis seperti relay, blower, dan pompa.

Dari perspektif karakteristik sistem, algoritma histeresis menawarkan kinerja dinamis yang superior dalam menjaga stabilitas proses. Aulia et al. (2025:64) dalam penelitian tentang optimasi suhu tanaman

karnivora menggunakan metode *Time Proportional Control Hysteresis* berbasis IoT mendefinisikan logika histeresis sebagai teknik pengendalian yang menggunakan dua batas nilai, yaitu batas atas dan batas bawah, untuk menjaga agar sistem tetap stabil. Pendekatan dengan dua batas nilai ini memastikan bahwa aktuator tidak akan terus-menerus berubah status hanya karena fluktuasi kecil pada pembacaan sensor, yang sering terjadi pada sistem kontrol on-off konvensional. Thobroni et al. (2024:278) menambahkan bahwa keuntungan menggunakan metode histeresis adalah kinerja dinamis yang sangat baik dan kemampuan untuk mengontrol nilai setpoint yang diinginkan, yang dalam konteks penelitian mereka berhasil mempercepat proses pembuatan pupuk kompos dari 30–40 hari menjadi hanya 21 hari. Kombinasi antara kemampuan menjaga stabilitas sistem dan fleksibilitas dalam menentukan rentang setpoint menjadikan algoritma histeresis sebagai pilihan yang optimal untuk berbagai aplikasi kontrol proses, mulai dari budidaya tanaman hingga pengolahan limbah organik.



Gambar 2.1 Kurva Histeresis
Sumber: Brown (2024)

Kurva histeresis yang ditunjukkan pada Gambar 2.1 menggambarkan hubungan antara input (E) dan output sistem (D), di mana terdapat dua jalur

berbeda saat nilai input meningkat dan menurun. Pada kurva tersebut terlihat adanya batas atas (*upper threshold*) dan batas bawah (*lower threshold*) yang membentuk loop histeresis. Titik D_s menunjukkan kondisi jenuh (saturation), sedangkan D_r merupakan titik respons sistem sebelum mencapai kondisi maksimum. Adanya perbedaan jalur ini menunjukkan bahwa sistem tidak langsung berubah status pada satu titik yang sama, melainkan menunggu hingga melewati batas tertentu. Inilah yang menyebabkan sistem menjadi lebih stabil dan tidak sensitif terhadap noise atau fluktuasi kecil.

3. *Internet of Things*

Mambang (2021:4) mendefinisikan konsep *Internet of Things* (IoT) sebagai sebuah paradigma teknologi yang menerjemahkan suatu objek fisik agar memiliki kemampuan untuk mentransfer data melalui sebuah jaringan tanpa memerlukan interaksi langsung dari manusia ke manusia, maupun interaksi dari manusia ke komputer. Kemampuan untuk beroperasi secara otonom ini mengubah benda-benda konvensional di sekitar kita menjadi perangkat cerdas yang dapat mengumpulkan, mengolah, dan mengirimkan informasi secara seketika (*real-time*). Dalam arsitektur *Smart Building* atau fasilitas kampus modern, IoT bertindak sebagai fondasi utama yang memungkinkan otomatisasi pada berbagai instrumen kelistrikan.

Lebih lanjut mengenai operasional ekosistem ini, Erwin et al. (2023:14) menjelaskan bahwa IoT merupakan konsep yang menghubungkan berbagai perangkat fisik secara langsung ke internet untuk berkomunikasi dan berbagi data secara bebas. Keberhasilan sistem komunikasi dan pertukaran

data ini ditopang oleh sebuah ekosistem yang terpadu. Erwin et al. (2023:14) menegaskan bahwa komponen-komponen utama yang membangun ekosistem IoT tersebut tidak berdiri sendiri, melainkan saling terintegrasi, yang secara garis besar meliputi sensor sebagai pengumpul data, perangkat terhubung sebagai medium fisik, jaringan komunikasi sebagai jalur transmisi nirkabel, serta *platform* dan aplikasi pengguna untuk memvisualisasikan data.

a. Mikrokontroler ESP32

Komponen komputasi utama dalam ekosistem *Internet of Things*, Sukowati et al. (2024:11) menjelaskan bahwa ESP32 merupakan lini mikrokontroler yang dikembangkan oleh Espressif Systems di Shanghai, Tiongkok, sebagai generasi penerus dari modul ESP8266 yang sebelumnya telah populer digunakan. Transisi ke modul generasi baru ini membawa pembaruan yang signifikan dari sisi arsitektur perangkat keras. Lebih lanjut mengenai spesifikasi tersebut, Sukowati et al. (2024:11) memaparkan bahwa ESP32 beroperasi sebagai mikrokontroler 32-bit berbasis *System on a Chip* (SoC) yang mengintegrasikan fitur Wi-Fi dan *Bluetooth* dengan tingkat konsumsi daya yang sangat rendah. Peningkatan performa pada modul ini juga ditandai dengan ketersediaan inti CPU yang lebih tangguh, konfigurasi pin *General Purpose Input/Output* (GPIO) yang lebih ekstensif, serta dukungan terhadap protokol *Bluetooth Low Energy* (BLE). Berkat spesifikasi komprehensif tersebut, ESP32 dinilai sangat ideal untuk bertindak sebagai pusat kendali (*edge computing*) pada sistem otomasi cerdas yang menuntut efisiensi daya sekaligus kecepatan

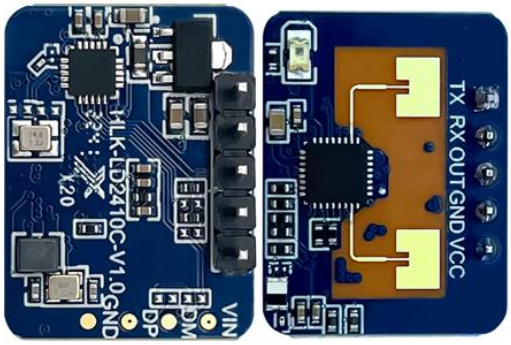
transmisi data nirkabel.. Bentuk fisik dari mikrokontroler ESP32 yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.2 dibawah ini.

Gambar 2.2 Mikrokontroler ESP32
Sumber: Sukowati et al. (2024:15)

b. Sensor Radar Gelombang Mikro (HLK-LD2410C)

Darsiman & Rabbika (2025:9) menjelaskan bahwa HLK-LD2410C merupakan perangkat modul yang menggunakan teknologi radar untuk mendeteksi keberadaan manusia, di mana kemampuannya dalam membedakan subjek hidup membuatnya jauh lebih akurat dibandingkan sensor deteksi gerakan konvensional. Secara teknis, keunggulan tingkat akurasi ini bersumber dari arsitektur gelombang yang digunakannya. Shenzhen Hi-Link Electronic (2022:3) dalam dokumen spesifikasi resminya mendefinisikan HLK-LD2410C sebagai modul sensor presensi manusia bersensitivitas tinggi yang beroperasi pada pita frekuensi gelombang mikro 24 GHz. Modul ini mengadopsi prinsip kerja *Frequency-Modulated Continuous Wave* (FMCW) untuk memancarkan gelombang radar secara berkelanjutan guna mengidentifikasi target manusia di dalam ruang yang telah dipetakan. Bentuk fisik dari sensor

HLK-LD2410C yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.3 dibawah ini.



Gambar 2.3 Sensor HLK-LD2410C
Sumber: Shenzhen Hi-Link Electronic (2022:6)

Lebih lanjut mengenai sensitivitasnya, Shenzhen Hi-Link Electronic (2022:3) memaparkan bahwa berbeda dengan solusi tradisional yang hanya peka terhadap objek yang bergerak aktif, teknologi FMCW pada sensor ini mumpuni dalam mendeteksi pergerakan mikro statis (*fretting*), seperti saat manusia sedang duduk, berdiri, atau berbaring. Kemampuan mendeteksi okupansi dengan jarak jangkauan hingga 5 meter dan sudut deteksi yang luas ini menjadikan sensor HLK-LD2410C sangat ideal untuk memantau kehadiran penghuni ruangan. Dalam konteks otomasi sistem pendingin ruangan, data dari sensor radar ini bertindak sebagai parameter *input* yang presisi untuk mencegah perangkat AC bekerja secara sia-sia ketika ruangan dalam keadaan kosong tanpa penghuni.

c. Sensor Suhu DHT22

Sensor DHT22 adalah modul instrumen digital yang berfungsi untuk mengakuisisi data parameter suhu dan kelembapan ruangan secara

simultan dengan tingkat presisi yang tinggi. Alfarizi (2025:10) menjelaskan bahwa sensor DHT22 merupakan modul sensor temperatur dan kelembapan yang memiliki tiga kaki utama sebagai antarmuka fisik, yaitu pin untuk VCC, DATA, dan GND. Konfigurasi kaki-kaki tersebut memfasilitasi integrasi yang efisien dengan mikrokontroler melalui komunikasi satu jalur (*single-bus*), sehingga mempermudah proses transmisi data termal secara digital tanpa memerlukan sirkuit pengkondisi sinyal tambahan.

Mengenai performa dan karakteristik operasionalnya, sensor ini dirancang untuk bekerja secara stabil pada berbagai rentang tegangan dan kondisi lingkungan yang bervariasi. Laoh et al. (2024:95) dalam penelitiannya memaparkan bahwa sensor suhu dan kelembapan DHT22 beroperasi dalam rentang tegangan 3 hingga 5,5 V DC, dengan waktu respons pembacaan data sekitar 2 detik. Rentang pengukuran suhu yang mampu dicakup oleh modul ini sangat luas, yakni antara -40°C hingga 80°C, dengan tingkat ketelitian sekitar ±0,5°C serta resolusi pembacaan mencapai 0,5°C. Kemampuan sensor DHT22 dalam menyediakan data secara *real-time* dengan akurasi tinggi sangat krusial dalam sistem pemantauan berbasis *Internet of Things* (IoT) agar informasi lingkungan dapat divisualisasikan secara langsung melalui aplikasi *mobile*. Keunggulan teknis ini menjadikannya komponen yang sangat ideal untuk mendeteksi fluktuasi suhu ruangan sebagai dasar pengambilan keputusan

pada sistem kendali otomatis. Bentuk fisik dari sensor DHT22 yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.4 dibawah ini.

Gambar 2.4 Sensor DHT22
Sumber: Laoh et al. (2024:95)

d. Modul *Infrared Transmitter* KY-005

Modul *Infrared Transmitter* KY-005 merupakan komponen elektronika krusial yang bertindak sebagai aktuator nirkabel antara mikrokontroler dan perangkat pendingin ruangan (AC). Ishari & Chandra (2023:2012) menjelaskan bahwa *Infrared Transmitter* KY-005 adalah perangkat yang dirancang khusus untuk mengirimkan sinar inframerah dalam sistem pengendalian jarak jauh. Keunggulan utama dari modul KY-005 ini terletak pada kemampuannya untuk berintegrasi secara mulus dengan mikrokontroler, sehingga memungkinkan sistem *Internet of Things* mengambil alih fungsi kendali manual secara terprogram. Dengan ukuran yang ringkas, perangkat ini dapat dengan mudah diletakkan berhadapan dengan *receiver* AC tanpa memakan banyak ruang. Bentuk fisik dari modul *infrared transmitter* KY-005 yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.5 dibawah ini.

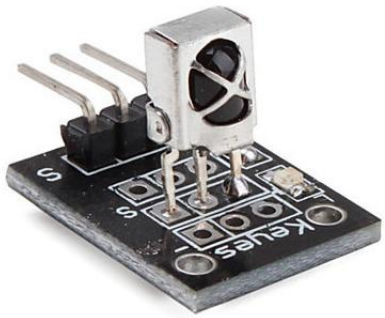
Gambar 2.5 Modul IR Transmitter KY-005
Sumber: Ishari & Chandra (2023:2013)

Secara teknis, proses transmisi sinyal pada modul ini didasarkan pada pancaran gelombang elektromagnetik tak kasat mata. Rozak et al. (2023:25) menguraikan bahwa KY-005 adalah modul pemancar inframerah dalam seri sensor komunikasi IR yang bekerja berdasarkan prinsip dasar *Light Emitting Diode* (LED). Modul ini memancarkan radiasi inframerah secara stabil pada frekuensi tertentu, yakni 38 kHz, dengan sudut pancaran efektif sebesar 20 derajat di sekitarnya. Melalui penyesuaian sinyal pada frekuensi 38 kHz tersebut, perangkat ini mampu mereplikasi kode protokol bawaan pabrik penyedia AC. Hal ini memastikan bahwa seluruh perintah komputasi dari aplikasi *mobile* seperti penyalaan, pemadaman, maupun transisi algoritma *Hysteresis* dapat dieksekusi secara presisi layaknya *remote* fisik bawaan pabrik.

e. Modul *Infrared Receiver* KY-022

Modul *Infrared Receiver* KY-022 merupakan modul penerima sinyal inframerah yang digunakan untuk membaca atau menerima pancaran sinyal dari perangkat *remote control*. Modul ini berfungsi sebagai penerima sinyal inframerah yang kemudian diteruskan ke

mikrokontroler agar dapat diproses sebagai data digital. Dalam sistem pengendalian perangkat elektronik, *IR Receiver* dapat digunakan untuk menangkap kode perintah dari *remote* asli sehingga kode tersebut dapat dijadikan acuan dalam proses pengendalian perangkat. Bentuk fisik dari modul *IR Receiver* KY-022 yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.6 dibawah ini.



Gambar 2.6 Modul IR Receiver KY-022
Sumber: Aris et al. (2025:37)

Aris et al. (2025:39) menjelaskan bahwa sistem monitoring *Air Conditioner* berbasis ESP32 dapat menggunakan sensor *infrared* KY-005 sebagai transmitter dan KY-022 sebagai receiver. *Infrared* pada sistem tersebut digunakan untuk mengirimkan sinyal kode on/off ke perangkat AC. Hal ini menunjukkan bahwa KY-022 dapat digunakan sebagai komponen penerima sinyal inframerah dalam sistem kontrol AC berbasis mikrokontroler.

Dalam penelitian ini, modul *IR Receiver* KY-022 digunakan sebagai komponen pendukung pada proses *cloning* sinyal inframerah dari *remote* asli AC. Mekanisme utama pengendalian AC tetap dilakukan menggunakan protokol inframerah bawaan *library* melalui *IR Transmitter* KY-005. Namun,

apabila protokol bawaan *library* tidak kompatibel dengan tipe AC tertentu, maka KY-022 digunakan untuk menerima dan merekam sinyal dari *remote* asli AC. Data sinyal yang diterima kemudian diproses oleh ESP32 dan dikirimkan ke *backend* untuk disimpan sebagai data *raw infrared*. Data tersebut selanjutnya dapat digunakan kembali sebagai alternatif perintah kontrol AC melalui *IR Transmitter* KY-005.

4. Protokol Komunikasi

a. MQTT

Dalam arsitektur *Internet of Things* (IoT), transmisi data antari-nstrumen memerlukan jalur komunikasi yang efisien dan andal. Rahmat et al. (2022:5) menjelaskan bahwa MQTT, yang merupakan kependekan dari *Message Queuing Telemetry Transport*, adalah sebuah protokol pesan berbasis sistem penerbitan dan berlangganan (*publish/subscribe*) yang telah diakui sebagai standar oleh *International Organization for Standardization* (ISO). Secara fungsional, MQTT bertindak sebagai protokol utama yang memfasilitasi transmisi data untuk menyampaikan pesan secara dua arah, baik dari pusat layanan (*server*/*broker*) menuju perangkat ujung (*client*), maupun sebaliknya. Mekanisme ini memungkinkan mikrokontroler pada setiap ruangan untuk terus bertukar data sensor dan status aktuator secara aktual tanpa membebani lalu lintas jaringan.

b. *WebSocket*

Sebagai pelengkap dari protokol komunikasi dalam arsitektur sistem pengontrolan cerdas, *WebSocket* berperan vital dalam memfasilitasi komunikasi yang instan pada sisi antarmuka pengguna. Merdekawan & Sari (2022:193) mendefinisikan *WebSocket* sebagai sebuah teknologi yang memungkinkan terjadinya sesi komunikasi interaktif dua arah (*full duplex*) antara pengguna dengan *server*. Dengan memanfaatkan *WebSocket API*, pengguna dapat mengirimkan perintah dan menerima respons secara langsung tanpa harus terus-menerus melakukan siklus permintaan balasan (*polling*) kepada *server* maupun perangkat IoT. Mekanisme komunikasi yang berkelanjutan ini sangat krusial untuk memastikan bahwa pembaruan data pembacaan sensor suhu dan status aktuasi AC dapat divisualisasikan secara *real-time* dan responsif di layar perangkat pengguna tanpa adanya jeda pemuatan ulang.

Lebih lanjut mengenai perkembangan implementasinya, Alviando et al. (2023:855) menjelaskan bahwa protokol ini pada awalnya dikembangkan di bawah standar HTML5 dan secara spesifik dirancang untuk mengoptimalkan komunikasi pada *web browser*. Namun, seiring berjalannya waktu dan tingginya eskalasi kebutuhan mobilitas pengguna, teknologi *WebSocket* kini telah diadaptasi secara luas untuk dapat berjalan pada perangkat bergerak (*mobile device*), termasuk integrasinya dengan bahasa pemrograman Android dan Swift. Dalam konteks rancang bangun sistem ini, keandalan konektivitas *WebSocket* lintas platform tersebut

dimanfaatkan secara penuh untuk menjembatani aliran data antara *server backend* dengan aplikasi *mobile native* yang dikembangkan menggunakan *framework* Flutter.

5. *Framework*

a. Express.js

Dalam pengembangan *backend* sebuah sistem, pemilihan kerangka kerja yang stabil sangat menentukan performa pertukaran data. Ramdhan et al. (2023:116) dalam penelitiannya menjelaskan bahwa Express.js merupakan salah satu teknologi *framework* yang digunakan untuk membangun API dengan menggunakan bahasa pemrograman JavaScript. Sebagai sebuah *web application framework* berbasis Node.js, Express menyediakan fitur yang handal untuk mendukung operasional aplikasi baik pada platform *web* maupun perangkat bergerak (*mobile*) secara efisien.

Karakteristik utama yang membuat teknologi ini unggul dibandingkan kerangka kerja lainnya adalah struktur arsitekturnya yang minimalis. Sutara & Gunawan (2024:20) menekankan bahwa *framework* Express.js dikenal sebagai perangkat lunak yang ringan karena tidak memerlukan banyak dependensi tambahan dalam proses instalasi dan pengembangannya. Penggunaan pola desain yang fleksibel pada Express memungkinkan pengembang untuk mengatur alur logika program secara bebas, sehingga sangat ideal digunakan untuk membangun sistem kontrol otomatis yang membutuhkan skalabilitas tinggi dan respon data yang

cepat. Integrasi antara Express.js dan protokol komunikasi seperti MQTT menjadikan lapisan pengolahan data pada sistem ini mampu menangani lalu lintas data sensor dari mikrokontroler menuju aplikasi pengguna secara optimal

b. Flutter

Affan et al. (2026:432) menjelaskan bahwa Flutter merupakan sebuah *Software Development Kit* (SDK) dari Google yang memungkinkan pengembangan aplikasi lintas platform dari satu basis kode dengan arsitektur berbasis *Widget Tree*, *Rendering Engine* Skia, dan *Framework Layer*. Melalui arsitektur ini, pengembang dapat menyusun antarmuka aplikasi dengan sangat fleksibel karena setiap elemen visual dalam aplikasi dianggap sebagai sebuah *widget* yang saling bersarang. Pendekatan tersebut mempermudah kustomisasi tampilan tanpa mengorbankan konsistensi performa pada berbagai platform yang berbeda.

Lebih lanjut mengenai pemanfataannya, Prameswari & Putro, (2026:607) menegaskan bahwa Flutter merupakan *framework open-source* yang dikembangkan oleh Google untuk membangun aplikasi *mobile cross-platform* pada sistem operasi Android dan iOS melalui satu basis kode tunggal. Teknologi ini sangat cocok diimplementasikan dalam proyek berbasis *Internet of Things* karena dukungannya yang luas terhadap integrasi protokol komunikasi seperti MQTT. Hal ini memungkinkan aplikasi untuk menampilkan data sensor secara *real-time* melalui antarmuka yang responsif dengan tingkat performa tinggi yang mendekati

aplikasi *native*. Kemampuan tersebut sangat krusial dalam sistem kontrol AC guna memastikan sinkronisasi data antara perangkat keras dan aplikasi pengguna berjalan tanpa kendala.

6. MySQL

Dalam arsitektur sistem informasi berbasis *Internet of Things* (IoT), penyimpanan riwayat aktuasi dan data sensor memerlukan sistem manajemen yang terstruktur dan responsif. Siregar et al. (2024:230) mendefinisikan MySQL sebagai salah satu sistem manajemen basis data relasional yang bersifat *open-source* dan telah diaplikasikan secara luas di berbagai perusahaan untuk keperluan pengolahan data secara efisien. Konsep relasional pada MySQL memungkinkan sistem *backend* untuk mengelola tabel-tabel informasi yang saling terhubung seperti tabel riwayat suhu, status presensi ruangan, dan catatan waktu nyala-mati AC secara terpusat, aman, dan terorganisir.

Lebih lanjut mengenai keunggulannya dalam tahap pengembangan sistem, Dachi & Suhada (2025:50) menjelaskan bahwa MySQL menjadi instrumen favorit bagi banyak pengembang dikarenakan kemudahannya dalam proses instalasi serta konfigurasi sistem. Dachi & Suhada (2025:50) juga menekankan bahwa MySQL menawarkan performa baca-tulis (*read-write*) data yang sangat cepat serta didukung oleh komunitas pengguna yang berskala besar. Keandalan dalam menangani proses manipulasi data (seperti menambah atau membaca log harian) dengan kecepatan dan konsistensi tinggi ini menjadikan MySQL sebagai basis data yang sangat ideal untuk

mencatat riwayat pemantauan secara berkelanjutan dari perangkat ESP32 maupun intervensi pengguna melalui aplikasi *mobile*.

7. Flowchart

Flowchart merupakan salah satu instrumen visual yang sangat penting dalam tahap perancangan, dokumentasi, maupun evaluasi sebuah sistem. Menurut Khairunnisa et al. (2023:88), *flowchart* adalah alat yang penting untuk peningkatan proses. Dengan menyediakan representasi grafis, *flowchart* ini membantu tim proyek untuk mengidentifikasi berbagai elemen proses dan memahami keterkaitan di antara berbagai langkah.

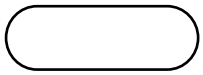
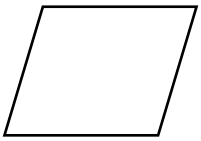
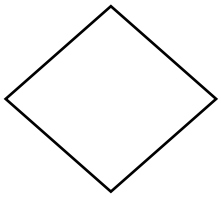
Dalam implementasinya, *flowchart* dapat diklasifikasikan ke dalam dua jenis, yaitu *flowchart* sistem dan *flowchart* program. Trisnanto (2024:100) menjelaskan bahwa *flowchart* sistem adalah diagram alur yang menggambarkan langkah-langkah atau proses yang terjadi dalam suatu sistem secara keseluruhan. Pemodelan ini membantu analis untuk melihat bagaimana aliran data dan kontrol bergerak antar komponen dalam ekosistem sistem tersebut.

Di sisi lain, Khairunnisa et al. (2023:87) mendefinisikan bahwa *flowchart* program adalah cara diagramatik untuk merepresentasikan algoritma yang digunakan untuk menghasilkan solusi dari suatu masalah. Melalui *flowchart* ini, alur logika komputasi dapat dipetakan secara terperinci langkah demi langkah sebelum kode program benar-benar dituliskan.

Dalam penyusunan sebuah *flowchart*, digunakan simbol grafis khusus yang memiliki makna operasional masing-masing. Setiap simbol

merepresentasikan instruksi spesifik, mulai dari titik awal proses, aliran data, pengambilan keputusan, hingga titik akhir. Adapun rincian simbol-simbol pada penelitian ini dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol *Flowchart*

Simbol	Nama	Fungsi
	Terminator	Digunakan untuk menunjukkan titik awal atau titik akhir dari sebuah program atau proses.
	Aliran Data / Flowline	Digunakan untuk menunjukkan arah aliran instruksi dari satu proses ke proses lainnya.
	Input/Output	Digunakan untuk menunjukkan *input* data atau *output* hasil dari sebuah program atau proses.
	Proses / Process	Digunakan untuk menunjukkan suatu tindakan atau pengolahan yang dilakukan dalam proses.
	Keputusan / Decision	Digunakan untuk menunjukkan titik pengambilan keputusan berdasarkan kondisi yang menghasilkan dua kemungkinan arah (Ya / Tidak).
	Dokumen	Digunakan untuk menunjukkan dokumen atau informasi yang diproses dalam alur kerja

Sumber: Khairunnisa et al. (2023:90)

8. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) merupakan salah satu bahasa pemodelan visual yang penting dalam proses perancangan, dokumentasi, dan pengembangan sistem perangkat lunak berbasis objek. Menurut Saputra et al. (2023:140), UML digunakan untuk memvisualisasikan, menspesifikasikan, membangun, dan mendokumentasikan suatu sistem perangkat lunak. Melalui pendekatan grafis, UML membantu pengembang dalam menggambarkan struktur dan perilaku sistem secara lebih terarah sebelum sistem masuk ke tahap implementasi.

Dalam penerapannya, Rafi et al. (2024:690) menjelaskan bahwa model UML digunakan untuk mendokumentasikan, merancang, dan mengomunikasikan desain sistem perangkat lunak. Pemodelan ini membantu menyamakan pemahaman antara pengembang, pengguna, dan pemangku kepentingan mengenai fungsi serta alur sistem yang akan dibangun.

Di sisi lain, Gunawan et al. (2023:52) menjelaskan bahwa tujuan utama penggunaan UML adalah membantu pengembang perangkat lunak dalam memvisualisasikan, mendokumentasikan, dan memahami rancangan sistem. Melalui UML, hubungan antarkomponen, alur aktivitas, interaksi antar objek, serta struktur data dapat dipetakan secara sistematis.

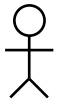
Dalam penelitian ini, UML digunakan dalam bentuk *Use Case* Diagram, *Activity* Diagram, *Sequence* Diagram, dan *Class* Diagram untuk menggambarkan fungsi, alur proses, interaksi, dan struktur Sistem Kontrol AC Multi-Ruang berbasis *Internet of Things*.

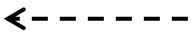
a. *Use Case* Diagram

Use Case Diagram merupakan salah satu diagram dalam UML yang digunakan untuk menggambarkan fungsi sistem berdasarkan sudut pandang pengguna atau pihak di luar sistem. Menurut Saputra et al. (2023:145), *Use Case* Diagram menunjukkan berbagai aktivitas yang dilakukan oleh sistem dengan berfokus pada apa yang dapat dikerjakan sistem, bukan pada cara sistem melaksanakan aktivitas tersebut. Pemodelan ini membantu pengembang memahami interaksi yang terjadi antara pengguna dan sistem.

Gunawan et al. (2023:55) menjelaskan bahwa tujuan *Use Case* Diagram adalah menggambarkan fitur-fitur utama yang dibutuhkan pengguna ketika menggunakan sistem. Melalui diagram ini, kebutuhan fungsional dapat ditampilkan dalam bentuk hubungan antara aktor dan *use case* sehingga ruang lingkup serta layanan yang disediakan sistem menjadi lebih mudah dipahami.

Tabel 2.2 Simbol *Use Case* Diagram

Simbol	Elemen	Keterangan
	Aktor	Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan use case.
	Use Case	Abstraksi dan interaksi antara sistem dan aktor.
	Assocation	Abstraksi dari penghubung antara aktor dengan use case.
	Generalisasi	Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case.

Simbol	Elemen	Keterangan
<<include>> 	Include	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya.
<<extend>> 	Extend	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi,

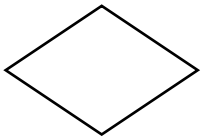
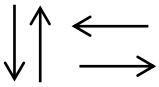
Sumber: Gunawan et al. (2023:56)

b. *Activity* Diagram

Activity Diagram merupakan salah satu diagram UML yang digunakan untuk menggambarkan urutan aktivitas dalam suatu proses. Menurut Saputra et al. (2023:156), *Activity* Diagram berfokus pada aktivitas-aktivitas yang terjadi dalam satu proses dan menunjukkan hubungan ketergantungan antara satu aktivitas dengan aktivitas lainnya. Pemodelan ini membantu pengembang memahami urutan proses, percabangan keputusan, serta aktivitas yang dapat dijalankan secara bersamaan.

Gunawan et al. (2023:64) menjelaskan bahwa tujuan *Activity* Diagram adalah memvisualisasikan alur kerja yang terjadi dalam suatu sistem atau proses bisnis serta mengidentifikasi bagian yang dapat dioptimalkan untuk meningkatkan efisiensi. Dalam penelitian ini, *Activity* Diagram digunakan untuk menggambarkan proses login, otomatisasi logika kendali AC, kontrol manual AC, dan *cloning* sinyal IR.

Tabel 2.3 Simbol *Activity* Diagram

Simbol	Elemen	Keterangan
	Activity	Memperlihatkan bagaimana masing masing kelas antarmuka saling berinteraksi satu sama lain.
	Action	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
	Initial Node	Menunjukkan titik awal dari diagram aktivitas
	Activity Final Node	Menunjukkan titik akhir dari diagram aktivitas
	Decision	Digunakan untuk menggambarkan suatu keputusan/tindakan yang harus diambil pada kondisi tertentu.
	Line Connector	Digunakan untuk menghubungkan satu simbol dengan simbol lainnya.

Sumber: Gunawan et al. (2023:65)

c. *Sequence* Diagram

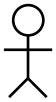
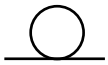
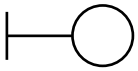
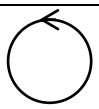
Sequence Diagram merupakan salah satu diagram UML yang digunakan untuk menggambarkan interaksi antara objek-objek dalam suatu sistem secara berurutan. Menurut Gunawan et al. (2023:60), *Sequence* Diagram menunjukkan aliran logika suatu fungsi melalui urutan interaksi antar objek. Diagram ini membantu pengembang memahami proses komunikasi yang terjadi antara pengguna, antarmuka, sistem, dan basis data.

Saputra et al. (2023:151) menjelaskan bahwa *Sequence* Diagram merupakan diagram interaksi yang menunjukkan cara suatu operasi dijalankan, pesan yang dikirimkan, serta waktu pelaksanaannya. Objek-

objek yang terlibat disusun berdasarkan waktu sehingga urutan pengiriman dan penerimaan pesan dapat terlihat secara jelas.

Selain itu, Gunawan et al. (2023:60) menjelaskan bahwa tujuan *Sequence* Diagram adalah menggambarkan urutan pesan atau pemanggilan antar objek serta memperlihatkan interaksi yang terjadi dalam waktu yang berbeda. Dalam penelitian ini, *Sequence* Diagram digunakan untuk menggambarkan proses login, otomatisasi logika kendali AC, kontrol manual AC, dan *cloning* sinyal IR.

Tabel 2.4 Simbol *Sequence* Diagram

Simbol	Elemen	Keterangan
	Aktor	Menggambar orang yang sedang berinteraksi dengan sistem.
	Entity Class	Menggambarkan hubungan yang akan dilakukan.
	Boundary Class	Menangani komunikasi antar lingkungan sistem.
	Control Class	Menggambarkan penghubung antar boundary dengan tabel
	A Focus of Control & A Life Line	Menggambarkan tempat mulai dan berakhirnya message.
	A Message	Menggambarkan pengiriman pesan.

Sumber: Gunawan et al. (2023:60)

d. *Class* Diagram

Class Diagram merupakan salah satu diagram UML yang digunakan untuk menggambarkan struktur kelas dan hubungan antarkelas dalam suatu sistem. Menurut Gunawan et al. (2023:57), *Class* Diagram

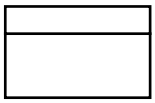
merepresentasikan kelas-kelas yang terdapat dalam sistem beserta hubungan yang terbentuk di antara kelas tersebut. Pemodelan ini membantu pengembang memahami susunan data, atribut, dan operasi yang diperlukan dalam pembangunan sistem.

Saputra et al. (2023:146) menjelaskan bahwa *Class* Diagram bersifat statis karena menggambarkan hubungan yang terdapat di antara kelas, bukan perubahan yang terjadi ketika kelas-kelas tersebut berinteraksi. Melalui pemodelan statis tersebut, struktur sistem dapat divisualisasikan secara jelas sebelum diterapkan ke dalam kode program maupun basis data.

Di sisi lain, Gunawan et al. (2023:57) menjelaskan bahwa tujuan *Class* Diagram adalah memberikan gambaran mengenai objek-objek yang terdapat dalam sistem serta hubungan di antara objek tersebut. Dalam penelitian ini, *Class* Diagram digunakan untuk menggambarkan struktur tabel users, ruangan, dan *log_history* beserta atribut dan hubungan yang digunakan pada Sistem Kontrol AC Multi-Ruang berbasis *Internet of Things*.

Tabel 2.5 Simbol *Class* Diagram

Simbol	Elemen	Keterangan
	Garis lurus (Generalization)	Menunjukan hubungan objek anak (descendent) dan induk (ancestor) dalam hal berbagai perilaku dan struktur datanya.
	Nary Association	Suatu upaya untuk menghindari asosiasi yang melebihi 2 objek.

Simbol	Elemen	Keterangan
	Class	Suatu himpunan dari objek-objek dalam sistem, yang kemudian berbagi atribut dan operasi yang persis sama.
	Collaboration	Berupa urutan aksi-aksi dalam sistem agar menghasilkan sebuah hasil yang terukur.
	Realization	Sebuah operasi yang benar-benar dilakukan oleh objek dalam sistem.
	Dependency	Suatu hubungan pada perubahan yang terjadi dalam independent yang mempengaruhi elemen yang tidak mandiri.
	Association	Bagian yang menghubungkan objek yang satu dengan yang lainnya.

Sumber: Gunawan et al. (2023:57)

9. *Black Box Testing*

Black Box testing merupakan metode evaluasi perangkat lunak yang secara khusus difokuskan pada uji fungsionalitas sistem. Pratama et al. (2023:561) menjelaskan metode ini bekerja dengan mengabaikan struktur internal dan logika kode dari perangkat lunak, sehingga perhatian dan evaluasi semata-mata ditujukan pada antarmuka, serta kesesuaian antara *input* dan *output* dari *software* tersebut. Sejalan dengan hal tersebut, Abdillah et al. (2023:235) menyatakan pendekatan ini diaplikasikan untuk menguji perangkat lunak yang telah dibangun, baik berupa pengujian pada unit-unit kecil maupun pada hasil yang telah terintegrasi secara keseluruhan guna memastikan fungsi perangkat lunak berjalan dengan baik. Secara praktis, tujuan utama dari metode ini adalah untuk memvalidasi operasional sistem secara menyeluruh, sehingga dapat dipastikan bahwa aplikasi yang

dikembangkan dapat berjalan sesuai dengan spesifikasi fungsional yang telah ditentukan.

Tingkat keberhasilan pengujian hasil *Black Box Testing* dapat dihitung menggunakan rumus persentase keberhasilan. Rumus ini digunakan untuk membandingkan jumlah *test case* yang berhasil dijalankan dengan total keseluruhan *test case* yang diuji. Adapun rumus persentase keberhasilan pengujian bisa dilihat pada persamaan 2.1 di bawah ini.

$$Persentase\ Keberhasilan = \frac{Jumlah\ Test\ Berhasil}{Total\ Test\ Case} \times 100\% \quad (2.1)$$

Pada persamaan 2.1, *Jumlah Test Berhasil* merupakan jumlah skenario pengujian yang menghasilkan keluaran sesuai dengan hasil yang diharapkan, sedangkan *Total Test Case* merupakan jumlah keseluruhan skenario pengujian yang dilakukan. Pada penelitian ini, rumus tersebut digunakan untuk menghitung tingkat keberhasilan pengujian aplikasi *mobile* berdasarkan jumlah skenario *Black Box Testing* yang berhasil dibandingkan dengan total skenario pengujian yang dilakukan.

B. Kajian Empiris

Penelitian Muzzaki et al. (2025) mengimplementasikan sistem kontrol AC otomatis menggunakan ESP8266, sensor PIR, dan aplikasi Blynk. Sistem berskala ruang kelas ini terbukti mampu menghemat daya listrik hingga 7,68 kWh per hari, atau memangkas biaya operasional sebesar 33,33%. Senada dengan hal tersebut, Nugroho (2025) juga mengembangkan sistem otomasi AC berbasis Arduino Uno dan sensor PIR untuk merespons okupansi di lingkungan kampus. Integrasi sistem ini menghasilkan penghematan energi mingguan antara

17% hingga 24% dibandingkan AC konvensional, tanpa mengurangi kenyamanan termal pengguna.

Dalam konteks pemerataan suhu, Prasetyo et al. (2024) merancang kendali AC menggunakan ESP32 yang dilengkapi empat sensor suhu DHT22 dan *infrared proximity*. Sistem ini mampu mendeteksi penghuni dengan akurasi di atas 95% dan secara otomatis mengaktifkan mode *swing* saat terjadi perbedaan suhu ruangan. Pendekatan ini mempercepat waktu pemerataan suhu hingga 10–15 menit, serta berhasil menjaga kenyamanan termal pada rentang 23°C–25°C dengan kendali jarak jauh responsif melalui aplikasi Blynk.

Lebih lanjut, Qornain et al. (2026) merancang prototipe sistem kontrol AC otomatis berbasis ESP8266, sensor DHT22, dan sensor PIR dengan aktuasi inframerah menggunakan pustaka IRac.h untuk mengontrol unit AC konvensional secara *retrofit*. Sistem tersebut mengimplementasikan logika kontrol adaptif yang mengaktifkan AC pada suhu normal (24°C) ketika ruangan panas dan berpenghuni, serta menurunkan suhu ke mode dingin (20°C) ketika deteksi gerakan terus-menerus mengindikasikan ruangan ramai, dengan estimasi penghematan energi sebesar 25%.

Pada skala gedung yang lebih besar, Chauhan (2025) melakukan studi kasus integrasi IoT dalam sistem HVAC di gedung perkantoran komersial (10.000 m²) dan gedung akademik kampus (5.000 m²) dengan arsitektur sensor nirkabel, *gateway* lokal, dan platform analitik *cloud*. Penelitian ini menerapkan *occupancy-based control* dan *demand-controlled ventilation*, menghasilkan penghematan energi HVAC sebesar 18–22% dengan peningkatan kenyamanan

penghuni dari 74% menjadi 88%. Namun, sistem tersebut menggunakan sensor PIR konvensional yang rentan kesalahan deteksi saat penghuni dalam kondisi diam.

Berdasarkan dari beberapa kajian diatas, penelitian terdahulu mayoritas masih berfokus pada ruangan tunggal dengan logika *on/off* dasar. Penggunaan sensor gerak konvensional juga rentan keliru saat penghuni diam, serta antarmukanya masih sangat bergantung pada platform pihak ketiga seperti Blynk. Oleh karena itu, penelitian ini hadir merancang Sistem Kontrol AC Multi-Ruang menggunakan radar gelombang mikro HLK-LD2410C yang lebih presisi dalam mendeteksi keberadaan penghuni.

Sebagai penyempurnaan lanjutan, penelitian ini mengimplementasikan algoritma *Hysteresis* di sisi *backend* untuk mengatur perpindahan mode pendinginan (*Turbo/Normal*) secara halus, guna mencegah keausan kompresor akibat siklus aktuasi yang terlalu cepat. Arsitektur komunikasi data dibangun secara mandiri menggunakan protokol MQTT dan *WebSocket* berbasis Express.js. Seluruh data tersebut kemudian divisualisasikan melalui aplikasi *mobile native* berbasis *framework* Flutter untuk mendukung *monitoring* multi-ruang dan rekapitulasi riwayat operasional secara aktual.

C. Kerangka Berpikir

Penelitian ini didasari oleh permasalahan inefisiensi konsumsi energi pada penggunaan AC di ruang kelas kampus. Mobilitas kegiatan perkuliahan yang tinggi sering menyebabkan AC tetap menyala meskipun ruangan kosong, sementara pengendalian masih bergantung pada *remote* fisik dan belum

didukung sistem pemantauan terpusat. Kondisi tersebut dapat menimbulkan pemborosan listrik, meningkatkan beban kerja kompresor, serta menyulitkan pengelola fasilitas dalam memantau penggunaan AC secara *real-time*. Oleh karena itu, diperlukan sistem kontrol AC otomatis yang adaptif terhadap suhu ruangan dan keberadaan manusia melalui pengembangan Sistem Kontrol AC Multi-Ruang berbasis *Internet of Things*.

Sistem ini dirancang untuk memantau dan mengontrol AC pada ruang kelas menggunakan ESP32, sensor HLK-LD2410C untuk mendeteksi keberadaan manusia, sensor DHT22 untuk membaca suhu, IR Transmitter KY-005 sebagai aktuator, serta IR Receiver KY-022 untuk proses *cloning* sinyal inframerah. Algoritma *Hysteresis* diterapkan pada backend untuk mengatur perpindahan Mode Turbo dan Mode Normal berdasarkan suhu ruangan, sedangkan timer delay 15 menit digunakan untuk mematikan AC ketika ruangan tetap kosong. Pemantauan dan pengendalian sistem dilakukan secara *real-time* melalui aplikasi mobile berbasis Flutter. Sistem yang dihasilkan diharapkan dapat membantu efisiensi penggunaan AC, memperpanjang usia pakai perangkat, dan mendukung penerapan *Smart Campus* di Universitas PGRI Madiun. Kerangka berpikir penelitian ini divisualisasikan melalui diagram pada Gambar 2.7.

Rumusan Masalah

1. Bagaimana merancang dan membangun Sistem Kontrol AC Multi-Ruang berbasis *Internet of Things* dengan algoritma *Hysteresis* terintegrasi aplikasi *mobile*?
2. Bagaimana mengimplementasikan sistem kontrol ac multi-ruang berbasis *internet of things* dengan algoritma *hysteresis* terintegrasi aplikasi *mobile*?
3. Bagaimana hasil pengujian dan evaluasi sistem kontrol ac multi-ruang berbasis *internet of things* dengan algoritma *hysteresis* terintegrasi aplikasi *mobile*?

↓

Model Penelitian

Penelitian ini menggunakan metode pendekatan Prototyping. Proses pengembangannya dimulai dari analisis kebutuhan sistem, perancangan prototype, evaluasi *prototype*, pengkodean sistem secara komprehensif, pengujian integrasi, serta tahap akhir berupa implementasi pada lingkungan target.

↓

Pengembangan Sistem

Sistem kontrol AC multi-ruang berbasis IoT ini dibangun meggunakan mikrokontroler ESP32, sensor radar HLK-LD2410C, dan sensor DHT22. Proses aktuasi menggunakan IR Transmitter KY-005 dan *cloning* sinyal menggunakan IR Receiver KY-022. Pengolahan logika *Hysteresis* dilakukan pada *backend* Express.js, dan data divisualisasikan serta dikendalikan secara *real-time* menggunakan aplikasi mobile berbasis Flutter dengan komunikasi MQTT dan WebSocket.

↓

Implementasi

Sistem kontrol AC multi-ruang berbasis IoT ini akan diimplementasikan di lingkungan akademis kampus Universitas PGRI Madiun, dengan fokus pada ruang kelas sebagai upaya optimalisasi manajemen energi fasilitas gedung.

↓

Hasil Penelitian

Sistem kontrol AC multi-ruang berbasis IoT yang mampu menyediakan otomasi perpindahan mode suhu dan aktuasi daya secara presisi, serta monitoring *real-time* via aplikasi *mobile* guna mendukung terwujudnya infrastruktur *Smart Campus* yang hemat energi.

Gambar 2.7 Kerangka Berpikir