PISTON:
Journal of Technical Engineering:

Sistem Pengendalian dan Pemantauan Terpusat pada Perangkat IoT Terdistribusi

Eka Stephani Sinambela[1,a)], Frengki Simatupang[1], Gerry Italiano Wowiling[1], Marojahan Mula Timbul Sigiro[1], Istas Pratomo Manalu[1], Sari Muthia Silalahi[1], Pandapotan Siagian[1]

[1]*Computer Technology, Vocational Faculty, Institut Teknologi Del Toba, 22381, Indonesia*

E-mail: [a)] eka@del.ac.id

Received: 22 Juni 2024 *Revision: 25 Juli 2024* *Accepted: 29 Juli 2024*

Abstrak: Perkembangan pesat Internet of Things (IoT) telah memungkinkan berbagai aplikasi cerdas; namun, pengelolaan banyak perangkat IoT yang tersebar secara manual masih kurang efisien dan memakan waktu. Penelitian ini bertujuan untuk mengembangkan sistem pemantauan dan pengendalian terpusat untuk perangkat IoT yang tersebar dengan menggunakan arsitektur master-agent. Master berfungsi sebagai pusat kendali yang mengumpulkan data dari berbagai agent serta memungkinkan manajemen terpusat melalui API Gateway yang memfasilitasi komunikasi dan kontrol perangkat. Prototipe yang dikembangkan terdiri dari dua mode kontrol: kontrol manual untuk menyalakan dan mematikan lampu melalui antarmuka berbasis web, serta kontrol otomatis untuk pemantauan lingkungan menggunakan berbagai sensor. Hasil pengujian menunjukkan bahwa sistem ini mampu mengelola perangkat IoT secara real-time dengan efektif. Pengujian kontrol manual berhasil mengaktifkan dan menonaktifkan lampu secara jarak jauh. Pengujian kontrol otomatis untuk pemantauan tanaman mencatat suhu lingkungan stabil antara 44–46°C, kelembaban tanah pada 27%, variasi jarak sensor ultrasonik antara 2–15 cm, serta fluktuasi intensitas cahaya antara 29–120 Cd. Hasil ini membuktikan bahwa sistem dapat merespons perubahan lingkungan secara dinamis, seperti mengaktifkan pompa air saat kelembaban tanah rendah atau menyesuaikan pencahayaan berdasarkan data real-time. Dengan menerapkan arsitektur RESTful API dan komunikasi berbasis JSON, sistem ini menawarkan skalabilitas tinggi dan fleksibilitas dalam pengembangan jaringan IoT. Penelitian ini menyimpulkan bahwa sistem pengendalian dan pemantauan IoT secara terpusat meningkatkan efisiensi, fleksibilitas, serta kemudahan dalam pengelolaan perangkat, sehingga dapat diterapkan dalam berbagai bidang seperti smart home, pertanian cerdas, dan otomatisasi industri.

Kata Kunci: IoT, Kontrol Terpusat, Sistem Pemantauan, Arsitektur Master-Agent, API Gateway, Otomatisasi.

Abstract: *The rapid development of the Internet of Things (IoT) has enabled various smart applications; however, managing multiple distributed IoT devices manually remains inefficient and time-consuming. This research aims to develop a centralized controlling and monitoring system for distributed IoT devices using a master-agent architecture. The master serves as the control center, collecting data from multiple agents and enabling centralized management through an API Gateway that facilitates communication and device control. The prototype consists of two control modes: manual control for turning lights on and off via a web-based interface and automatic control for environmental monitoring using sensors. Experimental results indicate that the system effectively manages IoT devices in real-time. The manual control test demonstrated successful remote activation and deactivation of multiple lights. The automatic control test for plant monitoring recorded stable temperature values between 44–46°C, soil moisture at 27%, ultrasonic sensor distance variations between 2–15 cm, and light intensity fluctuations between 29–120 Cd. These results confirm that the system can respond to environmental changes dynamically, such as activating a water pump when soil moisture is low or adjusting lighting conditions based on real-time data. By implementing a RESTful API architecture and JSON-based communication, the system offers high scalability and flexibility for expanding IoT networks. The findings conclude that a centralized IoT control and monitoring system enhances efficiency, flexibility, and ease of device management, making it applicable to various domains such as smart home, smart agriculture, and industrial automation.*

Keywords: *IoT, Centralized Control, Monitoring System, Master-Agent Architecture, API Gateway, Automation.*

PENDAHULUAN

Istilah "*Internet of Things*" (IoT) terdiri atas dua bagian utama yaitu internet yang mengatur konektivitas dan things yang berarti objek atau perangkat [1]. IoT adalah sebuah gagasan dimana semua benda di dunia nyata dapat berkomunikasi satu dengan yang lain sebagai bagian dari satu kesatuan sistem terpadu yang menggunakan jaringan internet sebagai penghubung [2]. Konsep IoT dengan cara kerja mengacu pada 3 elemen utama pada arsitektur IoT, seperti barang fisik yang dilengkapi modul IoT, perangkat koneksi ke internet seperti modem dan *Router*, dan *cloud data center* tempat untuk menyimpan aplikasi beserta *database* [3]–[5]. Dalam arsitekturnya, IoT terdiri dari perangkat keras, perangkat lunak, dan web, karena perbedaan protokol antara perangkat keras dengan protokol web, maka diperlukan *embedded system* berupa *gateway* untuk menghubungkan dan menjembatani perbedaan protokol tersebut [6], [7]. Manfaat yang dimiliki IoT dalam kehidupan sehari-hari sangat berpengaruh besar, diantaranya manfaat yang dimiliki IoT dapat berguna sebagai monitoring lingkungan contohnya pengecekan kondisi air [8], sebagai pengelolaan infrastruktur contohnya MRT [9] yang dibuat untuk transportasi cepat, sebagai media sensor peralatan yang biasanya digunakan pada perusahaan pertambangan [10], dan masih banyak lagi.

Seperti yang kita ketahui, teknologi *Internet of Things* (IoT) telah menjadi bagian dari kehidupan sehari-hari. Misalnya sistem *smarthome* [11] yang memanfaatkan teknologi IoT. Secara tidak langsung, IoT telah menyediakan berbagai solusi untuk permasalahan di sekitar kita. Dalam beberapa tahun terakhir, IoT ditingkatkan dengan adanya sensor dan aktuator (*Things*). *Things* itu merupakan sesuatu yang bisa dikendalikan melalui internet. Dapat dikaitkan pada sistem IoT ada perangkat komputasi yang saling terkait, dimana gabungan sensor dan aktuator berfungsi untuk meningkatkan interaksi objek terhadap kondisi sekeliling atau terhadap objek lain.

Kebanyakan dalam sebuah perangkat IoT, kontrol dan pemantauan biasanya menggunakan kontrol parameter manual oleh manusia. Dalam implementasinya, biasanya dalam satu perangkat IoT memiliki satu mikrokontroler sehingga melakukan konfigurasi harus secara manual terhadap setiap perangkat IoT tersebut. Hal ini tentu akan memakan banyak waktu jika dalam suatu ekosistem memiliki beberapa perangkat IoT yang bervariasi. Dalam suatu ekosistem IoT, terdapat sebuah keluaran yang penting, yaitu data. Data dapat berupa nilai sensor atau status dari sebuah aktuator. Dalam pengolahan datanya, beberapa perangkat IoT menggunakan layanan pihak ke tiga, salah satu contohnya adalah *Thingspeak*. *Thingspeak* merupakan *platform open source* aplikasi IoT yang digunakan untuk menyimpan dan mengambil data dari sesuatu menggunakan protokol HTTP [12]–[14]. Salah satu kekurangan dari penggunaan layanan pihak ke tiga adalah akses terhadap data yang terbatas. Dalam pengaplikasian IoT, sebuah perangkat IoT akan terdiri dari beberapa sensor dan aktuator, dan sebuah mikrokontroler.

Untuk melakukan kontrol terhadap mikrokontroler, masih harus dilakukan penggantian secara manual pada *source code*, dan melakukan flash ulang ke mikrokontrolernya. Hal ini tentu akan konsumsi banyak waktu jika harus melakukan hal yang sama ke beberapa perangkat IoT. Didasari oleh permasalahan di atas, penulis mencoba untuk membangun sistem dengan konsep *agent* dan *master*, dimana master yang melakukan *monitoring* dan *controling* terhadap beberapa *agent*-nya. *Agent* adalah sebuah perangkat IoT yang terdiri dari mikrokontroler yang bertugas sebagai otak untuk proses pengendalian, sensor dan aktuator, salah satu contohnya adalah suatu pemantauan yang terdiri dari beberapa sensor [15]–[20]. Agent biasanya terdistribusi sesuai dengan kebutuhan dari pengguna. Master adalah sebuah aplikasi yang berfungsi untuk mengumpulkan data, mengelola dan melakukan kontrol terhadap agent. Dengan mengaplikasikan ini, pengguna dapat dengan mudah mengubah nilai acuan yang ada di dalam perangkat IoT melalui aplikasi master, sehingga tidak lagi harus melakukan pengubahan secara langsung di mikrokontroler. Nilai acuan adalah sebuah nilai yang menjadi salah satu tolak ukur dari mikrokontroler dalam mengatur aktuatornya.

Dalam pengerjaannya, *agent* akan secara berkala memberikan datanya ke aplikasi *master*. Kemudian, aplikasi *master* akan membuat *decision* dari data yang diterima dan *state* yang diberikan oleh pengguna. Kemudian melalui *decision* ini akan melakukan *controlling* terhadap perangkat IoT tersebut. *Master* juga akan menyimpan data dan menyajikan data tersebut ke pengguna. Berdasarkan hal tersebut, tim peneliti melakukan implementasi master dan *agent* dengan 2 jenis *agent* yang berbeda. *Agent* yang pertama berupa pengolahan sensor yang menggunakan lebih dari satu sensor dimana pengontrolannya diambil berdasarkan nilai sensor dan *state* yang dimasukkan oleh pengguna (otomatis) dimasukkan dalam studi kasus pemantauan tanaman sayuran. Jenis *agent* yang ke dua adalah melakukan kontrol terhadap beberapa lampu (*on/off*), pengontrolan ini dilakukan langsung oleh pengguna (manual). Pada penelitian ini, membahas tentang penyimpanan data yang telah dikumpulkan oleh perangkat IoT. Seperti yang kita ketahui penyimpanan perangkat IoT itu sangat kecil, jadi dibutuhkan sebuah media yang digunakan untuk megumpulkan data tersebut. API *Gateway* yang akan menjadi penghubung antara perangkat IoT dengan *database*, yang akan mengelola segala komunikasi antara perangkat

dengan basis data [21]–[23]. API gateway yang menjadi sebuah jembatan untuk melakukan CRUD (*Create, Read, Update and Delete*) dalam sistem. Dengan adanya sistem ini, diharapkan dapat membantu pengguna dalam memantau dan mengontrol beberapa device IoT yang terdistribusi dan bervariasi secara terpusat. Dengan adanya aplikasi berbasis *web* yang digunakan sebagai antar muka pengguna untuk memudahkan pengguna dalam memantau kondisi dari lingkungan dengan kemampuan penggunaan sensor secara fleksibel pada sistem yang dibangun.

Penelitian ini bertujuan penelitian ini untuk membangun sebuah sistem terpusat yang mampu mengendalikan dan memantau lebih dari satu perangkat IoT yang terdistribusi dengan menggunakan konsep *Master* dan *Agent*. Sistem master sebagai sistem utama akan digunakan untuk mengendalikan dan memantau beberapa *agent* (perangkat IoT). Dengan demikian, pengendalian dan pemantauan perangkat IoT yang memiliki sistem secara independen dan bervariasi dapat diatur secara terpusat melalui sebuah panel sistem master.

METODOLOGI

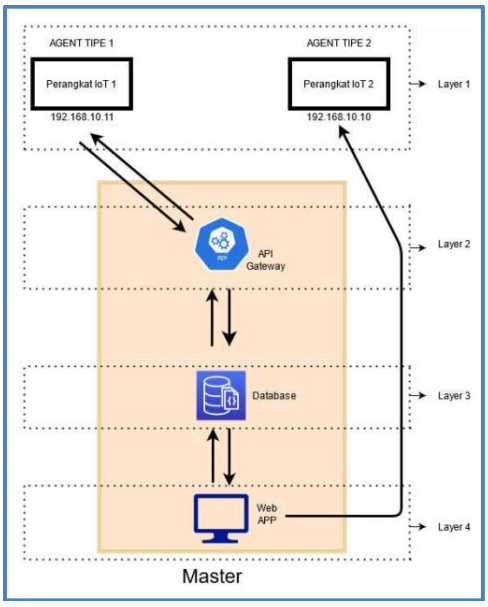
Analisis Masalah, Pemecahan Masalah dan Kebutuhan Perangkat

Pengontrolan dan pemantauan device IoT yang masih dilakukan secara manual dan terpisah menyebabkan kesulitan dalam konfigurasi dan integrasi antar perangkat [15], [24]. Sistem yang ada saat ini memerlukan konfigurasi ulang setiap kali ada penambahan device baru, sehingga kurang efisien dan rentan terhadap gangguan operasional. Untuk mengatasi masalah ini, diperlukan sistem sentral yang mampu mengelola komunikasi dengan setiap device IoT dalam satu jaringan agar lebih fleksibel dan terintegrasi. Implementasi komunikasi dua arah antara server dan device IoT dapat dilakukan menggunakan arsitektur RESTful API [20], [25] dengan format data JSON, yang ringan dan efisien untuk pertukaran informasi. Penggunaan metode POST [26] dalam pengiriman data lebih aman dibandingkan metode GET karena tidak menyimpan data di history browser dan mendukung berbagai jenis data. Sistem ini akan memudahkan pengguna dalam melakukan konfigurasi, monitoring, dan kontrol device IoT secara terpusat melalui aplikasi berbasis web sebagai User Interface. Wemos D1 dipilih sebagai mikrokontroler utama karena memiliki harga yang lebih terjangkau dibandingkan Raspberry Pi serta mampu menjalankan perintah sederhana dengan konektivitas WiFi [27], [28].

Pada studi kasus pemantauan tanaman, digunakan beberapa sensor seperti ultrasonik, DHT11, LDR, dan Soil Moisture, sedangkan untuk pengontrolan lampu hanya diperlukan LED tanpa sensor tambahan [29]–[31]. Web aplikasi dikembangkan menggunakan bahasa pemrograman PHP untuk memberikan interface yang user-friendly bagi pengguna dalam mengelola sistem. Dalam penyimpanan dan pengelolaan data, MongoDB dipilih sebagai database karena kemampuannya yang fleksibel dalam menangani data dengan struktur yang dinamis.

Perancangan Perangkat Lunak

Gambaran sistem yang akan dibangun terdiri dari 2 bagian umum yaitu *agent system* dan *master system*, sistem ini dibagi menjadi 4 layer.



Gambar 1. Gambaran umum sistem

Layer 1: Pada layer pertama pada sistem yang dibangun ini merupakan agent yang merupakan perangkat iot atau client-client yang dalam satu client terdiri dari mikrokontroler, sensor dan ada juga aktuatornya. Dapat dilihat dari Gambar 1 bagian Layar Agent merupakan bagian utama yang pertama menjelaskan tentang perangkat IoT yang akan dikontrol atau dimonitor oleh master. Dalam pembangunan sistem ini terdapat terdapat dua jenis client, dimana jenis client yang pertama adalah client yang akan dikontrol secara otomatis berdasarkan nilai sensor yang diterima oleh master. Nilai dari sensor akan dikirimkan secara berkala ke aplikasi master. Jenis client yang kedua adalah client yang akan dikontrol secara manual oleh administrator. Client dan master terhubung oleh koneksi wireless dengan protokol HTTP. Pada Client tipe kedua ini, aplikasi master melalui aplikasi web akan mengirimkan HTTP request kepada client melalui port 80. Client akan selalu melakukan listen melalui port 80. HTTP request ini berupa URL yang disertai data pin dan perintahnya. Berikut adalah contoh dari URL yang berisikan nomor pin dan perintah:

```
http://192.10.10.2/2/on/3/off
```

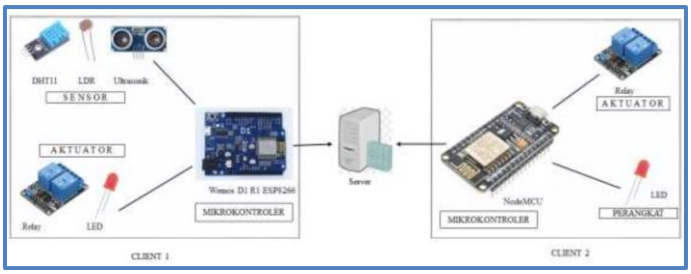
URL diatas terdiri dari IP address agent yang akan dikontrol (http://192.10.10.2), dan nomor pin serta perintah (2/on/3/off). Pada agent akan terdapat code yang akan membaca pin dan perintah sehingga agent dapat melakukan kontrol terhadap aktuatornya. Dengan perintah tersebut, agent akan menghidupkan pin nomor 2 dan mematikan pin nomor 3.

Layer 2: Sistem yang dibangun ini merupakan Server yang memiliki Gateway dan Database di dalamnya , dimana gateway disini yang menjadi gerbang komunikasi antara database dan device nya. API Gateway berfungsi sebagai satu-satunya akses dari setiap agent untuk melakukan create data baru di dalam basis data, dan juga melakukan pengontrolan otomatis terhadap agent. API dibutuhkan untuk mentranslasikan Bahasa pemograman yang ada di mikrokontroler (C), web (PHP) dan query database. Saat menerima file JSON dari agent, API Gateway akan mentranslasikan file JSON tersebut menjadi array yang bisa diproses oleh Bahasa pemograman PHP. Sebelum memasukkan ke dalam database, API Gateway terlebih dahulu melakukan pengecekan apakah jumlah data yang diterima sesuai dengan jumlah sensor yang terdapat dalam basis data, jika hal ini tidak sesuai, maka proses akan berhenti. Jika jumlah data dan sensor sama, maka API Gateway akan melakukan pengecekan terhadap status sensor. Terdapat 2 jenis status sensor. Status pertama yaitu active, yang berarti data yang diterima dari sensor tidak akan dilakukan pengubahan. Status kedua yaitu deactive, yang berarti pengguna tidak menginginkan data dari sensor tersebut, oleh karena itu setiap data yang masuk akan diganti menjadi nol (0). Setelah pengecekan selesai, maka API Gateway akan menambahkan data tanggal dan waktu ke dalam array kemudian data akan disimpan di dalam database. Untuk melakukan pengontrolan otomatis, API Gateway akan mengambil informasi nilai acuan sensor di basis data, kemudian melakukan perbandingan terhadap data yang diterima dari client dengan data acuan yang terdapat di basis data. Setelah melakukan perbandingan data, maka API Gateway kemudian akan membuat URL untuk melakukan kontroling terhadap agent. URL ini kemudian akan di curl menggunakan Bahasa pemograman PHP untuk mengontrol agent.

Layer 3: Pada layer tiga ini pada sistem yang dibangun merupakan database store sebagai tempat penyimpanan data yang diterima dari agent. **Layer 4:** Pada layer empat ini pada sistem yang dibangun merupakan User Interface yang digunakan sebagai antar muka pengguna untuk memantau, mengontrol dan memonitoring setiap sensor dan menampilkan data yang diterima dari sensor.

Sistem Master adalah aplikasi master yang terdiri dari 3 bagian utama yaitu, API Gateway, database, dan aplikasi web yang terdapat pada layer 2,3 dan 4. API Gateway akan menerima data dari setiap agent, kemudian API Gateway akan melakukan modifikasi terhadap datanya, menyimpan data ke dalam basis data, dan melakukan kontrol terhadap agent. Database akan menyimpan semua data, termasuk data setiap sensor, dan data informasi perangkat yang dimasukkan oleh pengguna. Kemudian Aplikasi web akan menampilkan data, sebagai antarmuka untuk pengguna melakukan CRUD (create, read, update, delete) terhadap data di database, melakukan kontroling secara langsung terhadap setiap agent.

Perancangan Perangkat Keras

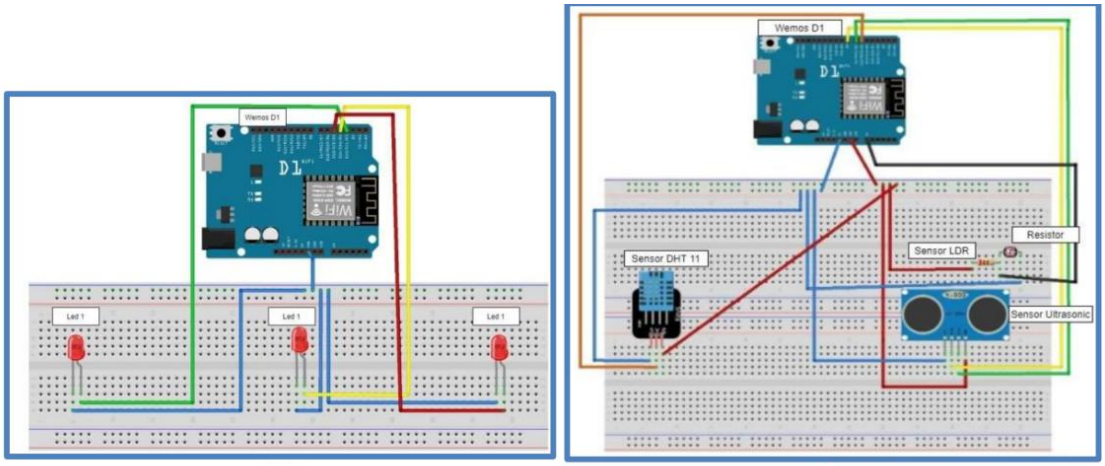


Gambar 2. Perancangan perangkat Keras

Pada Gambar 2 di atas, dapat dilihat bahwa sistem memiliki dua bagian utama, yaitu server sebagai pusat data, dan client sebagai pengumpul data. *Client* adalah sistem IoT yang bertugas untuk mengumpulkan data, baik melalui sensor ataupun status dari setiap aktuator. Client merupakan sistem tertanam tradisional yang memiliki sensor, aktuator, dan aktuator. Namun, pada hal ini setiap device harus memiliki kemampuan untuk terhubung kedalam jaringan wifi. Maka dari itu, developer akan menggunakan Wemos D1 R1 sebagai *microcontroller*. Wemos D1 R1 memiliki kemampuan untuk terhubung kedalam jaringan wifi karena didalamnya sudah terdapat *modul* ESP8266. ESP8266 merupakan sebuah modul dimana modul tersebut memungkinkan sebuah *microcontroller* untuk terhubung kedalam sebuah jaringan wifi. Server adalah pusat dari sistem yang kami bangun. Server akan menerima setiap data yang dikirim oleh setiap client dan melakukan pengolahan terhadap data tersebut (*create, read, update, delete*) pada database. Setelah melakukan pengolahan terhadap data, server juga dapat mengirimkan perintah terhadap setiap device, hal ini bergantung dari data yang diberikan dan masukan dari user terhadap client yang bersangkutan.

HASIL DAN PEMBAHASAN

Implementasi prototipe pengendali lampu LED dan Monitoring Tanaman



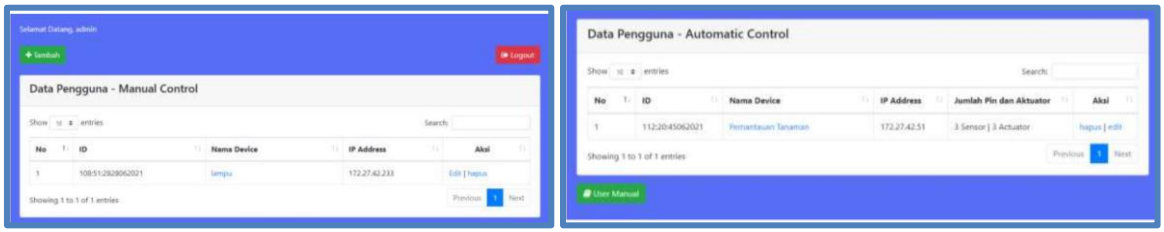
Gambar 3. Implementasi rangkaian prototipe (a) Sistem pengendali Lampu LED (b) Sistem Monitoring Tanaman

Implementasi rangkaian pada prototype pengontrolan LED hidup/mati Gambar 3(a) adalah menghubungkan lampu led dengan pin wemos . Dalam rangkaian terdapat 3 lampu , dimana kaki Negatif pada led akan terhubung dengan GND. Kaki positif pada lampu led 1 akan terhubung dengan pin D3. Kaki positif pada lampu led 2 akan terhubung dengan pin D4. Kaki positif pada lampu led 3 akan tergubung dengan pin D5.

Sementara itu, implementasi rangkaian pada Prototipe pemantauan tanaman Gambar 3 (b) yang menggunakan beberapa sensor adalah membuat penghubung antara pin . Pin yang ada pada sensor dihubungkan dengan menggunakan kabel *jumper*. Pada sensor DHT11 pin yang digunakan adalah pin VCC, ground, Data. Pin tersebut akan dihubungkan pada Wemos D1R1. Pin VCC dihubungkan menggunakan jumper berwarna biru ke pin 5 V sebagai tegangan positif terhadap sensor. Pin ground dihubungkan menggunakan jumper berwarna merah ke pin GND yang merupakan 0 V power supply. Pin tersebut menjadi sumber tegangan negatif sensor. Pin Data dihubungkan menggunakan jumper berwarna orange ke pin D6.

Pada sensor Ultrasonic pin yang digunakan adalah pin VCC, ground, Echo, data 1 dan data 2. Pin VCC dihubungkan menggunakan jumper berwarna biru ke pin 5 V. Pin ground dihubungkan menggunakan jumper berwarna merah ke pin GND. Pin data 1 atau biasa disebut sebagai trigger terhubung pada kabel jumper kuning ke pin D8. Pin ini berfungsi untuk membangkitkan sinyal ultrasonik. Pin data 2 atau biasa disebut pin echo terhubung pada kabel hijau ke pin D7. Pin ini berfungsi mendeteksi sinyal pantulan ultrasonik. Pada sensor LDR pin yang digunakan adalah pin VCC,ground dan A0. Pin VCC akan dihubungkan dengan menggunakan kabel jumper berwarna biru. Pin ground akan dihubungkan dengan menggunakan kabel jumper berwarna merah. Pin A0 akan dihubungkan dengan pin A0, diman pin A0 berfungsi untuk input dan output digital yang dapat mengubah sinyal analog menjadi nilai digital yang mudah diukur.

Implementasi Aplikasi Web



Gambar 4. Implementasi dashboard aplikasi web

Pada komponen layar utama Gambar 4, setelah pengguna berhasil melakukan proses autentikasi (login), sistem secara otomatis akan mengarahkan pengguna ke tampilan **dashboard**. Dashboard ini menampilkan tabel yang berisi daftar **agent** yang dikontrol secara manual maupun otomatis. Sesuai dengan studi kasus yang diimplementasikan, tabel **"Data Pengguna – Manual Control"** mencakup studi kasus pengontrolan **on/off** lampu. Pada tabel ini, terdapat beberapa atribut utama, yaitu **ID agent, nama perangkat, alamat IP (IP address), serta aksi** yang memungkinkan pengguna untuk mengedit atau menghapus agent yang terdaftar. Sementara itu, tabel **"Data Pengguna – Automatic Control"** digunakan untuk studi kasus **pemantauan tanaman**, yang mencakup atribut **ID agent, nama perangkat, jumlah pin, jumlah aktuator, serta aksi** untuk mengedit atau menghapus agent yang terdaftar. Selain itu, pada halaman dashboard juga terdapat menu **User Manual**, yang berfungsi sebagai panduan penggunaan sistem. Menu ini menyediakan informasi terkait prosedur penambahan agent dan dilengkapi dengan kode pemrograman yang harus diterapkan pada perangkat IoT sebelum didaftarkan ke dalam sistem.

Implementasi Menambah Agent

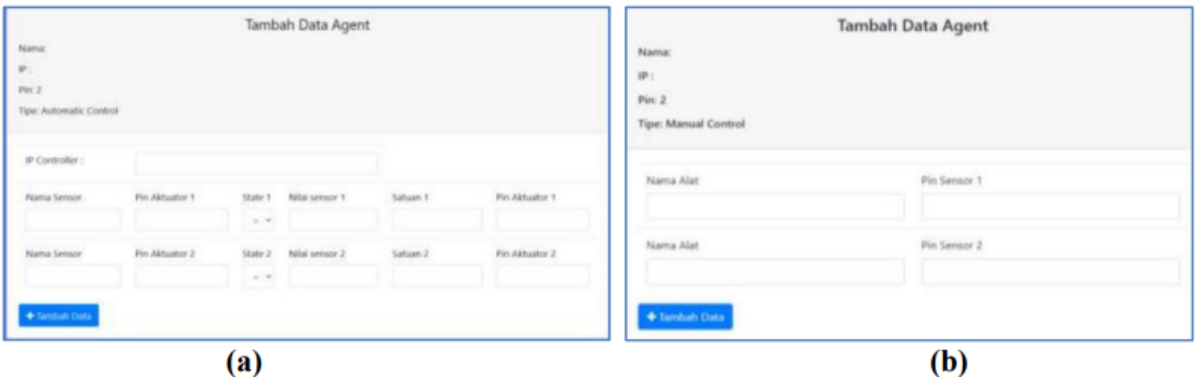
Pada komponen tambah agent, ketika pengguna memilih tombol "Tambah Agent", sistem akan menampilkan halaman formulir untuk menginput informasi terkait agent yang akan didaftarkan ke dalam sistem. Formulir ini mencakup beberapa parameter utama, yaitu nama agent, alamat IP (IP address), jumlah sensor yang digunakan, serta tipe agent, yang dapat diklasifikasikan sebagai kontrol manual atau kontrol otomatis. Pemilihan tipe kontrol ini akan menentukan bagaimana agent tersebut berinteraksi dengan sistem utama, baik melalui intervensi langsung pengguna maupun melalui otomatisasi berdasarkan data sensor yang dikumpulkan.

Selain itu, informasi yang diinput oleh pengguna pada halaman ini akan disimpan dalam database dan diintegrasikan dengan mekanisme komunikasi sistem, seperti RESTful API, untuk memastikan bahwa setiap agent yang ditambahkan dapat beroperasi secara optimal dalam jaringan IoT yang telah dikembangkan. Tampilan antarmuka dari halaman ini disajikan pada Gambar 5, yang menggambarkan struktur form pengisian data agent secara sistematis. Desain formulir ini bertujuan untuk mempermudah pengguna dalam proses registrasi perangkat IoT ke dalam sistem serta meningkatkan efisiensi pengelolaan agent dalam lingkungan IoT yang kompleks.

Gambar 5. Implementasi menambah agent

Implementasi Menambah Agent dengan Tipe Manual dan Automatic Control

Komponen layar tambah agent untuk yang dikontrol secara otomatis Gambar 6 (a) , jika user memilih tipe dari agent yang ditambahkan adalah "Automatic Control" maka selanjutnya user akan diarahkan ke halaman baru untuk memberi informasi lebih terperinci mengenai sensor apa saja yang akan digunakan, dengan mengisi nama-nama dari sensor, pin dari masing-masing sensor, nilai dari masing-masing sensor dan juga pin dari aktuator yang akan dihubungkan ke masing-masing sensor nantinya saat melakukan konfigurasi. Komponen layar tambah agent untuk yang dikontrol secara manual Gambar 6 (b), jika user memilih tipe dari agent yang ditambahkan adalah "Manual Control" maka selanjutnya user akan diarahkan ke halaman baru untuk memberi informasi mengenai nama dari alat dan pin sensornya.



Gambar 6. Implementasi menambah agent (a) Tipe Automatic Control (b) Tipe Manual Control

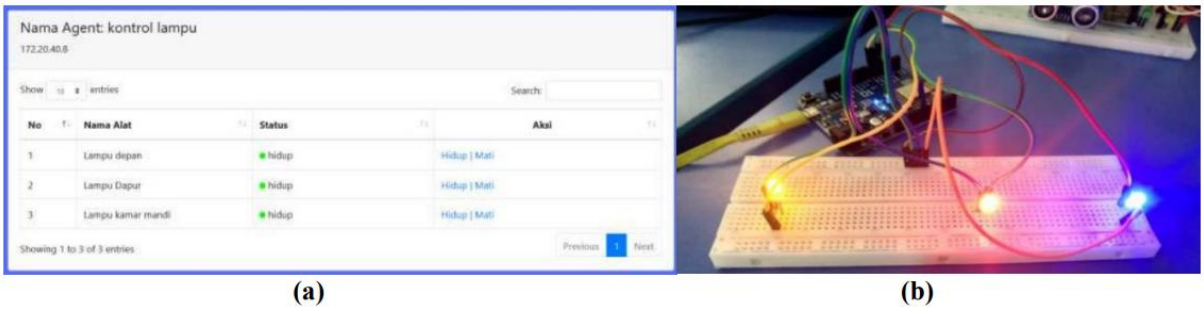
Komponen layar **edit** memungkinkan pengguna untuk menghapus atau mengedit agent yang telah terdaftar, di mana penghapusan agent juga akan menghapus seluruh perangkat yang terhubung dengannya. Selain itu, pada mode **Automatic Control**, pengguna dapat mengedit deskripsi agent, sementara pada mode **Manual Control**, pengguna dapat mengedit atau menghapus lampu yang telah didaftarkan.

Pengujian Prototype Device IoT Tipe Manual Control

Pada saat melakukan pengujian pada pengontrolan lampu, developer menggunakan 3 led pada satu device IoT yang dikontrol secara manual melalui web aplikasi dengan menekan aksi On/Off.

Tabel 1. Pengujian prototype device manual control

Tahapan Uji Coba	Hasil Uji Coba
Aksi ON/OFF lampu depan	Berhasil
Aksi ON/OFF lampu dapur	Berhasil
Aksi ON/OFF lampu kamar mandi	Berhasil



Gambar 7. Implementasi (a) Tampilan web control lampu (b) Prototype kondisi lampu

Gambar ini menunjukkan sistem pengontrolan lampu berbasis IoT. Gambar 7 (a): Tampilan antarmuka web yang digunakan untuk mengontrol status lampu secara manual. Terdapat daftar lampu dengan status hidup atau mati, serta opsi untuk mengubah statusnya. Gambar 7 (b): Rangkaian perangkat keras menggunakan mikrokontroler dan breadboard yang mengontrol lampu LED berdasarkan perintah dari antarmuka web. Sistem ini memungkinkan pengguna untuk mengontrol lampu secara jarak jauh melalui antarmuka berbasis web.

Pengujian Prototipe Device IoT Tipe Automatic Control

Tabel 2. Pengujian prototipe Device IoT automatic Control

No	Suhu (ºC)	Kelembaban Tanah (%)	Jarak sensor 1Ultrasonik	Intensitas Sensor LDR (Cd)
1	44	27.2	2	120
2	44	27.1	5	100
3	44	27.2	15	108
4	44	27.1	15	29
5	44	27	4	98
6	44	27	5	102
7	45	27	6	103
8	46	27	8	110

Pengujian ini bertujuan untuk mengevaluasi kinerja perangkat IoT tipe Automatic Control dalam memantau parameter lingkungan secara real-time. Parameter yang diuji meliputi suhu (°C), kelembaban tanah (%), jarak sensor ultrasonik (cm), dan intensitas cahaya dari sensor LDR (Cd). Berdasarkan hasil pengujian, suhu lingkungan cenderung stabil pada 44–46°C, sementara kelembaban tanah tetap konstan pada 27%.

Pengukuran jarak menggunakan sensor ultrasonik menunjukkan variasi antara 2 cm hingga 15 cm, yang mengindikasikan perubahan kondisi objek di sekitar sensor. Sementara itu, intensitas cahaya yang terdeteksi oleh sensor LDR berfluktuasi antara 29 Cd hingga 120 Cd, menunjukkan respons sensor terhadap perubahan pencahayaan di lingkungan pengujian. Dari hasil ini, sistem dapat secara otomatis menyesuaikan respons aktuator berdasarkan parameter yang terdeteksi, seperti pengaktifan pompa air jika kelembaban tanah rendah atau penyesuaian pencahayaan berdasarkan data dari sensor LDR. Hal ini menunjukkan bahwa prototype IoT tipe Automatic Control mampu bekerja secara efektif dalam melakukan pemantauan lingkungan secara otomatis dan real-time.

Pengujian Monitoring Tanaman/Sayuran

Pengujian Monitoring Tanaman/Sayuran akan memantau kondisi lingkungan tanaman secara real-time menggunakan perangkat IoT. Data yang diukur mencakup kelembaban udara (RH%), suhu (°C), intensitas cahaya (Cd), dan jarak (cm). Gambar 8 menunjukkan antarmuka monitoring sistem, di mana tabel bagian atas menampilkan data hasil sensor dalam bentuk angka dan waktu pencatatan. Setiap entri berisi nilai kelembaban udara, suhu, intensitas cahaya, serta jarak, yang terus diperbarui dalam interval tertentu. Bagian bawah menampilkan grafik pemantauan tanaman, yang menggambarkan tren perubahan parameter lingkungan. Grafik ini membantu pengguna dalam menganalisis pola perubahan kelembaban, suhu, pencahayaan, dan jarak dalam kurun waktu tertentu. Hasil pengujian menunjukkan bahwa suhu cenderung stabil di angka 44°C, kelembaban udara sedikit berfluktuasi, intensitas cahaya mengalami perubahan signifikan, dan jarak sensor mendeteksi variasi objek di sekitarnya. Dengan sistem ini, pengguna dapat melakukan pemantauan kondisi tanaman secara otomatis, yang memungkinkan tindakan korektif jika terjadi anomali, seperti penyiraman otomatis saat kelembaban rendah atau penyesuaian pencahayaan jika intensitas terlalu rendah.

Gambar 8. Hasil Monitoring Tanaman

KESIMPULAN

Sistem Pengendalian dan Pemantauan Terpusat pada Perangkat IoT Terdistribusi yang dikembangkan dalam penelitian ini berhasil mengatasi keterbatasan metode manual dalam mengendalikan dan memantau perangkat IoT yang tersebar dengan menerapkan model master-agent. Master berfungsi sebagai pusat kendali, sementara agent mengumpulkan data dan menjalankan perintah berdasarkan konfigurasi parameter. Implementasi API Gateway memungkinkan komunikasi efisien antara perangkat IoT dan basis data melalui proses CRUD serta mendukung kontrol otomatis terhadap agent. Prototipe sistem memiliki dua jenis kontrol: manual dan otomatis. Kontrol manual untuk menyalakan dan mematikan lampu melalui antarmuka web, yang terbukti berfungsi baik. Kontrol otomatis diterapkan untuk pemantauan lingkungan tanaman menggunakan

sensor suhu, kelembaban tanah, intensitas cahaya, dan sensor ultrasonik. Hasil pengujian menunjukkan suhu lingkungan stabil pada kisaran 44–46°C, kelembaban tanah tetap 27%, jarak yang terdeteksi oleh sensor ultrasonik bervariasi antara 2–15 cm, dan intensitas cahaya 29–120 Cd. Sistem menyesuaikan aktuator berdasarkan parameter yang terdeteksi, seperti pengaktifan pompa air saat kelembaban tanah rendah dan penyesuaian pencahayaan dari sensor LDR. Sistem juga menunjukkan waktu respons cepat dalam mengeksekusi perintah, memastikan keandalan pengendalian perangkat. Dengan arsitektur berbasis RESTful API dan komunikasi JSON, sistem ini fleksibel serta mudah diperluas untuk mendukung perangkat baru, sehingga berpotensi diterapkan dalam smart home, smart agriculture, dan industri.

DAFTAR PUSTAKA

[1] R. Abdmeziem and D. Tandjaoui, "Internet of Things: Concept, Building blocks, Applications and Challenges," 2014.

[2] M. Lombardi, F. Pascale, and D. Santaniello, "Internet of things: A general overview between architectures, protocols and applications," *Inf.*, vol. 12, no. 2, pp. 1–21, 2021, doi: 10.3390/info12020087.

[3] P. Sethi and S. R. Sarangi, "Internet of Things: Architectures, Protocols, and Applications," *J. Electr. Comput. Eng.*, vol. 2017, 2017, doi: 10.1155/2017/9324035.

[4] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: a review," *J. Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0268-2.

[5] T. Domínguez-Bolaño, O. Campos, V. Barral, C. J. Escudero, and J. A. García-Naya, "An overview of IoT architectures, technologies, and existing open-source projects," *Internet of Things (Netherlands)*, vol. 20, p. 100626, 2022, doi: 10.1016/j.iot.2022.100626.

[6] E. Simeoni *et al.*, "A Secure and Scalable Smart Home Gateway to Bridge Technology Fragmentation," *MDPI*, pp. 1–23, 2021, doi: https://doi.org/10.3390/s21113587.

[7] Y. M. Algani, "Integration of Internet Protocol and Embedded System On IoT Device Automation," *researchsquare*, pp. 1–14, 2021, doi: https://doi.org/10.21203/rs.3.rs-947704/v1 License:

[8] I. Essamlali, H. Nhaila, and M. El Khaili, "Advances in machine learning and IoT for water quality monitoring: A comprehensive review," *Heliyon*, vol. 10, no. 6, p. e27920, 2024, doi: 10.1016/j.heliyon.2024.e27920.

[9] M. Mohamed, K. Alosman, M. Mohamed King Abdul Aziz university, S. Arabia, and K. Alosman King Abdul Aziz university, "An IoT-Enabled Framework for Smart City Infrastructure Management," 2024.

[10] T. Zvarivadza *et al.*, "On the impact of Industrial Internet of Things (IIoT) - mining sector perspectives," *Int. J. Mining, Reclam. Environ.*, vol. 38, no. 10, pp. 771–809, 2024, doi: 10.1080/17480930.2024.2347131.

[11] C. Jin, J. Su, and W. Sun, "Design and implementation of smart home system based on IOS platform," vol. 24, no. June, p. 25, 2022, doi: 10.1117/12.2642582.

[12] P. Rajnandini, T. Tambe, V. Vishwakarma, and R. Ansari, "International Journal of Research Publication and Reviews Water Level Monitoring System Using IOT," vol. 3, no. 11, pp. 1159–1161, 2022.

[13] D. Nettikadan and S. Raj M S, "Smart Community Monitoring System using Thingspeak IoT Plaform," *Int. J. Appl. Eng. Res.*, vol. 13, pp. 13402–13408, Sep. 2018.

[14] J. Kandimalla and D. D. R. Kishore, "Web Based Monitoring of Solar Power Plant UsingOpen Source IOT Platform Thingspeak andArduino," *Int. J. Mod. Trends Sci. Technol.*, vol. 03, no. 4, pp. 16–21, 2017.

[15] D. Witczak and S. Szymoniak, "Review of Monitoring and Control Systems Based on Internet of Things," *Appl. Sci.*, vol. 14, no. 19, 2024, doi: 10.3390/app14198943.

[16] Z. Wu, K. Qiu, and J. Zhang, "A Smart Microcontroller Architecture for the Internet of Things," *Sensors (Switzerland)*, vol. 20, no. 7, pp. 1–17, 2020, doi: doi:10.3390/s20071821.

[17] G. M. Kapitsaki, A. P. Achilleos, P. Aziz, and A. C. Paphitou, "Sensoman: Social management of context sensors and actuators for IoT," *J. Sens. Actuator Networks*, vol. 10, no. 4, 2021, doi: 10.3390/jsan10040068.

[18] O. Arshi and S. Mondal, "Advancements in sensors and actuators technologies for smart cities: a comprehensive review," *Smart Constr. Sustain. Cities*, vol. 1, no. 1, p. 18, 2023, doi: 10.1007/s44268-023-00022-2.

[19] M. Mazo Jr. and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," 2010. doi: 10.48550/arXiv.1004.0477.

[20] N. Dipsis and K. Stathis, "A RESTful middleware for AI controlled sensors, actuators and smart devices," *J. Ambient Intell. Humaniz. Comput.*, vol. 11, no. 7, pp. 2963–2986, 2020, doi: 10.1007/s12652-019-01439-3.

[21] M. Bauer *et al.*, "The context API in the OMA next generation service interface," *2010 14th Int. Conf. Intell. Next Gener. Networks "Weaving Appl. Into Netw. Fabr. ICIN 2010 - 2nd Int. Work. Bus. Model. Mob. Platforms, BMMP 10*, no. May 2014, 2010, doi: 10.1109/ICIN.2010.5640931.

[22] F. Cirillo, G. Solmaz, E. L. Berz, M. Bauer, B. Cheng, and E. Kovacs, "A Standard-Based Open Source IoT Platform: FIWARE," *IEEE Internet Things Mag.*, vol. 2, no. 3, pp. 12–18, 2020, doi: 10.1109/iotm.0001.1800022.

[23] S. Jeong, S. Kim, and J. Kim, "City Data Hub: Implementation of Standard-Based Smart City Data Platform for Interoperability," *Sensors*, vol. 20, no. 23. 2020. doi: 10.3390/s20237000.

[24] T. Domínguez-Bolaño, V. Barral, C. J. Escudero, and J. A. García-Naya, "An IoT system for a smart campus: Challenges and solutions illustrated over several real-world use cases," *Internet of Things (Netherlands)*, vol. 25, no. February, p. 101099, 2024, doi: 10.1016/j.iot.2024.101099.

[25] R. Maurya, "Application of Restful APIs in IOT: A Review," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 2, pp. 145–151, 2021, doi: 10.22214/ijraset.2021.33013.

[26] A. A. Kumar and D. Tl, "Security measures implemented in RESTful API Development," vol. 07, no. September, pp. 105–112, 2024.

[27] S. Purba, M. Hariri, R. J. Banjarnahor, and S. N. Siregar, "LED Control System Using Arduino Wemos D1 R1 Based on Web Server Communication Via Internet of Things (IoT)," *Formosa J. Sci. Technol.*, vol. 2, no. 6, pp. 1397–1408, 2023, doi: 10.55927/fjst.v2i6.4436.

[28] P. R.Dinkir, Patnaik, "A Comparative Study of Arduino, Raspberry Pi and ESP8266 as IoT Development Board," *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 5, pp. 2350–2352, 2017.

[29] G. R. Choudhari, P. A. Dagale, I. S. Dashetwar, R. R. Desai, and A. A. Marathe, "IoT-based Smart Gardening System," *J. Phys. Conf. Ser.*, vol. 2601, 2023, doi: 10.1088/1742-6596/2601/1/012006.

[30] C. R. Gunawan, D. Ramadani, and F. Amir, "Monitoring System for Soil Moisture and Lighting in Decorative Plants," *Proc. 2nd Int. Conf. Sci. Technol. Mod. Soc. (ICSTMS 2020)*, vol. 576, no. Icstms 2020, pp. 332–335, 2021, doi: 10.2991/assehr.k.210909.074.

[31] A. R. A. Rashid, M. K. A. Azmi, W. M. Mukhtar, and N. A. M. Taib, "IoT-Integrated Smart Gardening System for Real-Time Monitoring and User-Controlled with Smart Film," *J. Telecommun. Electron. Comput. Eng.*, vol. 17, no. 1, pp. 11–17, 2025, doi: 10.54554/jtec.2025.17.01.002.