Fungsi Ganda Perangkat *IoT-Client* sebagai Kendali Aktuator dan Basis Layanan Melalui Komunikasi Paket Data JSON

Isa Albanna[1,*], Hendro Nugroho[2]
[1] Program Studi Sistem Informasi, FTETI, Institut Teknologi Adhi Tama Surabaya
[2] Program Studi Teknik Informatika, FTETI, Institut Teknologi Adhi Tama Surabaya
Email: [*]isaalbanna@itats.ac.id

DOI: https://doi.org/10.31284/j.jtm.2023.v4i2.4699
Received 12 July 2023; Received in revised 24 July 2023; Accepted 25 July 2023; Available online 31 July 2023
Copyright: ©2023 Isa Albanna, Hendro Nugroho
License URL: https://creativecommons.org/licenses/by-sa/4.0

Abstract

Internet of Things (IoT) is an integrated system for connecting between devices in the internet network. Data exchange in IoT must be managed properly to avoid errors in sending, interpreting and making decisions in IoT systems. The role of MQTT (message queuing telemetry transport) in IoT data management requires another form of innovation, because several MQTT platforms depend on the IoT building blocks. Through this research a communication system has been designed using JSON and micropython to provide two functionalities in the RPi IoT-Client. The first function is for actuator activation and the second function is as a service unit. The research method was carried out in five stages, namely the Raspberry Pi Pico W-based IoT instrument design stage, IoT network integration, interface programming preparation, back-end programming preparation and system testing. In translating JSON data via micropython, a decision will be made by the system for the activation process of the On-Off LED light actuator. On the service base function, the RPi IoT Client has a data insert function in the MySQL database engine. The results obtained from the preparation of the system, obtained the value of functional dualism can run with a synchronous system. Validation in LED light activation has an accuracy rate of around 88%, this is due to network loading when sending data continuously every 2 seconds. Through this technique, it is hoped that it can become an alternative to IoT data management besides using MQTT-brokers.

Keywords: *MQTT architecture; JSON data; Internet of Things; IoT-client Raspberry Pi Pico; Data communication;*

Abstrak

Internet of Things (IoT) merupakan sistem terintegrasi untuk menghubungkan antar perangkat dalam jejaring internet. Pertukaran data dalam IoT harus dikelola dengan baik untuk menghindari kesalahan pengiriman, interpretasi dan pengambilan keputusan dalam sistem IoT. Peran MQTT (message queuing telemetry transport) dalam manajemen data IoT memerlukan bentuk inovasi lain, karena beberapa platform MQTT memiliki kebergantungan terhadap instrumen penyusun IoT. Melalui penelitian ini telah dirancang sebuah sistem komunikasi menggunakan JSON dan micropython untuk memberikan dua fungsional dalam IoT-Client RPi. Fungsi pertama adalah untuk aktivasi aktuator dan fungsi kedua adalah sebagai unit pelayanan. Metode penelitian dilakukan dalam lima tahap, yaitu tahap perancangan instrumen IoT berbasis Raspberry Pi Pico W, integrasi jaringan IoT, penyusunan pemrograman antarmuka, penyusunan pemrograman back-end dan pengujian sistem. Pada penerjemahan data JSON melalui micropython akan didapatkan pengambilan putusan oleh sistem untuk proses aktivasi aktuator lampu LED On-Off. Pada fungsi basis layanan, RPi IoT Client memiliki fungsi insert data dalam mesin basis data MySQL. Hasil yang didapatkan dari penyusunan sistem tersebut, didapatkan nilai dualisme fungsional dapat berjalan dengan sistem sinkron. Validasi dalam aktivasi lampu LED memiliki tingkat akurasi sekitar 88%, hal ini disebabkan akibat pembebanan jaringan saat dilakukan pengiriman data secara

kontinu per 2 detik. Melalui teknik ini, diharapkan dapat menjadi alternatif manajemen data IoT selain menggunakan MQTT-broker.

Kata Kunci: Arsitektur MQTT; Data JSON; Internet of Things; IoT-client Raspberry Pi Pico; Komunikasi Data;

1. Pendahuluan

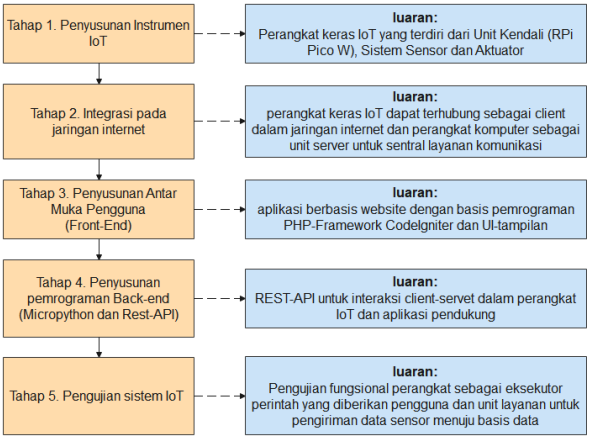
Internet of Things (IoT) dapat dipandang suatu sistem terintegrasi dengan melibatkan perangkat keras, aplikasi, perintah komputer dan jalur komunikasi data untuk menghubungkan antar alat agar dapat terhubung secara luas dalam jaringan internet. Bentuk perspektif lain dari IoT adalah M2M (*machine to machine*)[1]. Konsep M2M lahir untuk menjembatani adanya komunikasi dua arah antara mesin dengan mesin melalui protokol komunikasi data yang bersifat khusus dan spesifik[2]. Protokol komunikasi dalam M2M diantaranya adalah Lora[3], ZigBee[4], Bluetooth Low Energy (BLE) [5] dan TCP/IP[6]. Pada jenis protokol Lora, ZigBee, dan BLE memiliki lingkup yang cukup terbatas, karena data-transfer harus berada pada jalur komunikasi (*gateway*) satu jenis instrumen dan komunikasi tanpa melibatkan data internet. Pada studi kasus komunikasi TCP/IP dalam pengembangan IoT memiliki cakupan yang cukup luas, karena informasi dari komunikasi menggunakan jalur data internet yang memungkinkan adanya penggunaan instrumen heterogen dalam layer aplikasi [7]. Teknik komunikasi data IoT dalam jalur internet dibutuhkan peran unit *server* (layanan) sebagai pengolah transaksi data dalam sebuah sistem IoT dan *client* (klien) sebagai unit peminta layanan. Konsep layanan yang cukup komplek dalam ekosistem IoT, melahirkan adanya unit layanan jembatan yang disebut sebagai IoT-*broker* yang tergabung dalam MQTT (*Message Queuing Telemetry Transport)*[8] [9].

Penelitian terkait IoT selain pada bidang komunikasi, beberapa penelitian juga mengambil fokus pada instrumen pendukung IoT. Beberapa perangkat keras yang mampu mendukung IoT, pada umumnya memiliki perangkat yang mampu terhubung pada jalur TCP/IP seperti modul Wifi dan LAN-RJ45. Perangkat varian ESP8266, ESP32, modul Arduino-wifi dan Raspberry Pi Pico Tipe W (RPi-Pico W) [10], [11] merupakan bentuk instrumen yang mampu menjembatani antara sistem tertanam (*Embedded System*) dan komunikasi data internet melalui socket TCP/IP. Penggunaan MQTT sebagai *broker* dalam manajemen komunikasi data IoT terdapat aspek kurang efisien jika diimplementasikan dalam instrumen dalam kuantitas kecil [12]. Kebutuhan dalam membangun sistem berbasis MQTT membutuhkan instrumen server, jasa layanan MQTT, perangkat kendali yang terhubung sensor-aktuator. Pertimbangan aspek biaya jasa layanan, pengaturan bentuk frame data (dalam proses *subscribe-publish*) dan penyesuaian desain antar muka aplikasi menjadi landasan penggunaan sistem back-end IoT micropython yang terintegrasi dalam perangkat klien-IoT Raspberry Pi Pico W.

Pada penelitian ini difokuskan dalam pengerjaan implementasi JSON untuk mendukung fungsi ganda instrumentasi klien-IoT yaitu sebagai unit kendali aktivasi aktuator dan basis layanan melalui pemrograman *framework* micropython. Unit instrumen klien-IoT yang digunakan dalam penelitian adalah Raspberry Pi tipe Pico W (dapat disebut RPi Pico W). Tujuan dari penelitian ini adalah membuat dua fungsi RPi Pico W yang mana objek tersebut disebut sebagai IoT-*Client* dengan dua fungsi yaitu eksekutor perintah untuk menjalankan aktuator melalui instruksi yang diberikan oleh operator dalam antarmuka website. Fungsi yang kedua dari *IoT-Client* adalah sebagai unit layanan yang memiliki peran pengirim data sensor untuk proses perekaman data dalam mesin basis data. Harapan dari penelitian ini adalah penggunaan dua fungsi IoT-*Client* dapat menjadi media optimasi sistem perangkat keras untuk menunjang manajemen dan komunikasi data dalam sistem *Internet of Things* yang efisien, efektif dan dinamik.

2. Metode

Penelitian dilaksanakan dalam lima tahap, yaitu tahap penyusunan perangkat keras (instrumen IoT) yang terdiri dari papan kendali raspberry Pi Pico W dual-core Arm Cortex-M0, tahap integrasi perangkat instrumen IoT-*Client* dalam jaringan internet melalui peran moda klien-AP (*access point*), tahap pemrograman website *front-end* dengan menggunakan basis HTML dan CSS-Bootstrap, tahap penyusunan layanan *back-end* (Rest API) untuk menghubungkan, mengatur dan melakukan operasi penambahan data dalam mesin basis data MYSQL dan tahap pengujian fungsional sistem IoT. Adapun detail langkah-langkah dan target luaran yang didapatkan dalam masing-masing proses disajikan dalam diagram alir penelitian Gambar 1, dalam pemrograman sistem website digunakan bahasa pemrograman PHP yang terintegrasi dalam *framework* CodeIgniter 4.



Gambar 1. Tahap penelitian dualisme fungsional instrumen IoT-*Client* yang meliputi penyusunan, integrasi dan pengujian.

Luaran penyusunan perangkat keras IoT pada penelitian yang dikerjakan adalah menghasilkan perangkat keras yang mampu mengatur aksi aktuator dengan merujuk perintah pengguna melalui antarmuka web, membaca data sensor dan mengirimkan data sensor menuju mesin basis data. Perangkat keras Raspberry Pi Pico W memiliki perangkat wifi *single-band* 2.4GHz *wireless interfaces* (802.11n) yang nantinya digunakan sebagai jembatan komunikasi data antara struktur data sistem tertanam dan komunikasi internet. Penggunaan jaringan internet dalam penelitian digunakan sumber provider yang dibagikan melalui perangkat gawai. Proses pemanggilan domain atau server, dilakukan dengan memasukkan Alamat IP dari mesin layanan (*server*). Pada studi kasus pada penelitian terdapat dua macam pemanggilan perintah dalam mesin layanan oleh klien, permintaan tersebut adalah pemanggilan perintah oleh Instrumen IoT-*Client* (perangkat keras) dan pemanggilan perintah oleh operator (klien dengan menggunakan komputer). identifikasi Alamat IP nantinya digunakan sebagai akses website. Operator mengirimkan perintah sebagai bentuk representasi permintaan aktivasi aktuator atau permintaan untuk perekaan data dalam basis data (*insert data*). *User Interface* atau aplikasi antar muka pengguna, pada penelitian disusun dalam bentuk website untuk mendukung interaksi antara sistem IoT dan pengguna.Pada pemrograman *back-end* digunakan pemrograman python yang terangkum dalam micropython versi v1.20.0 (2023-04-26). Masing-masing blok pengerjaan penelitian akan dijelaskan secara detail dalam paparan tahap sebagai berikut:

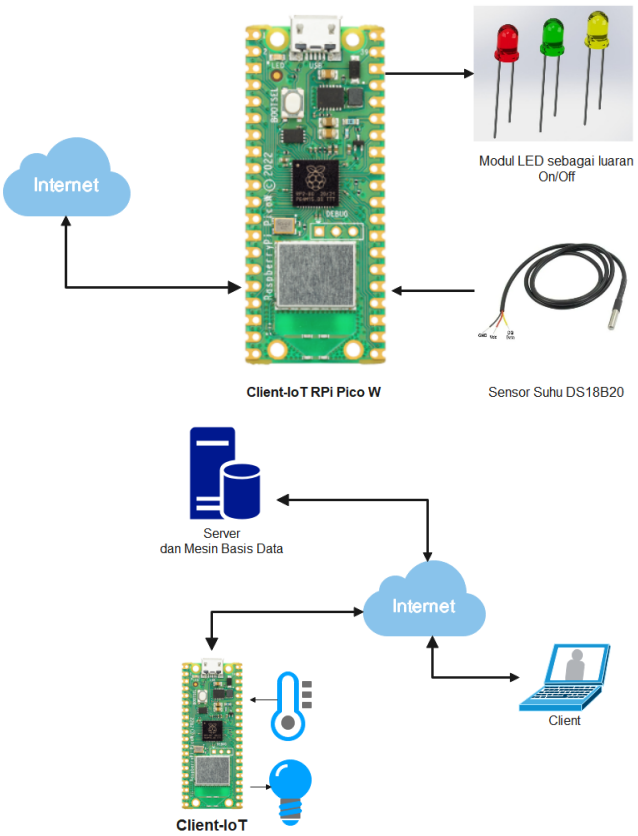
2.1 Tahap Pertama - Penyusunan Instrumen IoT

Instrumen IoT pada penelitian terdiri dari unit kendali utama, perangkat aktuator dan unit sensor suhu. Kendali utama atau IoT-*Client* pada penelitian digunakan Raspberry Pi tipe Pico W atau dapat disebut RPi Pico W. Pada RPi Pico W telah terintegrasi mesin kompilasi python yang tergabung dalam micropython v1.20.0 berukuran 1.4 MB dan pustaka pendukung akses GPIO. Perangkat aktuator berupa LED (*light emitting diode*) 3.3volt yang terhubung secara aktif-*HIGH*, artinya ketika logika digital memiliki level 3.3volt, maka LED tersebut akan menyala. Pada penelitian bentuk

perangkat keras luaran hanya digunakan sebuah LED yang nantinya dapat diganti dengan alat lainya yang lebih komplek dalam pengembangan lebih lanjut. Proses nyala dan mati dari lampu LED merupakan bentuk luaran kendali jarak jauh melalui perintah operator dalam jaringan IoT. Perangkat keras Raspberry Pi Pico akan mengidentifikasi data yang dikirim oleh operator dan diterjemahkan dalam data biner (satu-kosong) yang nantinya diteruskan dalam logika ON-OFF LED. IoT-Client dianggap berhasil dalam menerjemahkan perintah operator jika logika menyalanya LED sesuai dengan data permintaan operator. Pada perangkat keras sistem sensor, telah dipasang sensor suhu DS18B20. Perangkat tersebut terhubung dalam pin GPIO yang mendukung komunikasi one-wire. Raspberry Pi PICO W mampu mendukung komunikasi one-wire secara akurat untuk proses akuisisi data [13] dalam jaringan IoT. Perangkat sensor suhu DS18B20 merupakan unit input yang nantinya data suhu tersebut akan diambil, diolah dan dikirim untuk perekaman dalam tabel data mesin basis data yang terhubung dengan mesin server. Pada Gambar 2.a merupakan susunan IoT-*Client* dan sistem pelengkap untuk pemrosesan informasi melalui protokol TCP/IP. Micropython yang tertanam dalam RPi Pico W, memberikan dukungan berupa komputasi numerik untuk sparasi dan penerjemahan data JSON yang dikirimkan dari server menuju IoT-*Client*.

2.2 Tahap Kedua - Integrasi pada Jaringan Internet

Pada tahap integrasi internet, dapat diilustrasikan seperti pada Gambar 2b yaitu terdiri dari satu mesin *server*, dua *Client* dan layanan internet. Mesin server digunakan komputer yang didalamnya sudah terpasang mesin server Apache, basis data MySQL, penerjemah PHP 7.4 dan *framework* CodeIgniter 4. Perangkat keras server yang digunakan dalam penelitian memiliki spesifikasi sebagai berikut: prosesor Intel core I7, RAM 16 Giga dan instrumen jaringan. *Client* yang terhubung pada server dalam penelitian ini, dibatasi hanya dua unit *Client* yaitu IoT-*Client* (berupa perangkat keras) dan Operator-*Client* yang terhubung pada komputer. Operator akan mengakses tampilan antar muka yang diletakkan dalam server untuk mengatur tugas dari IoT-*Client*. Skenario dalam perintah dalam sistem IoT-Client dikirimkan dalam bentuk data JSON yang nantinya diterjemahkan oleh algoritma *text-identification* yang dituliskan melalui pemrograman micropython. Pengiriman data dari operator menuju ke IoT-Client memiliki Alamat ID yang tersinkron dengan relasi basis data dari daftar alat yang telah diregistrasikan. Penggunaan Alamat ID-Alat memiliki tujuan bahwa data yang dikirim dalam bentuk JSON tepat pada sasaran atau target alat yang akan dikendalikan atau diatur. ID perangkat keras yang telah diregistrasikan, nantinya akan dikirim dalam untaian data *key-value* dalam paket JSON.



(a) (b)

Gambar 2. a) Desain instrumen IoT-*Client* Raspberry Pi Pico W, b) Integrasi IoT *Client*-server dalam jaringan internet

2.3 Tahap Ketiga - Penyusunan Antar Muka Pengguna

Aplikasi yang berperan sebagai fungsional kendali dibuat pada domain website. Perancangan pada domain tersebut, membutuhkan standar pemrograman web berbasis PHP. Pemrograman web dirancang dengan arsitektur MVC (*Model View Controller*) yang tergabung dalam *Framework* CodeIgniter. Pada bentuk pemrograman CodeIgniter terdapat manajemen *route* atau pengalamatan URL yang nantinya dapat mengakses pada sisi *controller* program. Pemisahan view pada *framework* tersebut dapat memberikan struktur efisien dan integrasi *framework* lainya untuk penyusunan suatu tampilan antar muka. Pada penelitian ini digunakan *framework* Bootstrap V5. Sebagai bentuk kendali pada tampilan diletakkan bentuk Form-HTML untuk menangani data yang dikirim melalui metode POST/GET oleh *Client*.

2.4 Tahap Keempat - Penyusunan Pemrograman *Back-End*

Pemrograman *back-end* merupakan suatu bentuk *script* atau pemrograman yang berjalan dibalik layar untuk membantu dalam manajemen data, pengambilan keputusan, kendali dan pengelolaan *input-output.* Pada penelitian ini, pemrograman *back-end* terdapat dua jenis, yaitu program dalam sisi server (berupa REST-API) dan sisi IoT-*Client* (berupa penerjemahan paket data JSON). Pemrograman *back-end* sisi *server*, digunakan bahasa pemrograman PHP yang tergabung dalam *framework* CodeIgniter 4. REST-API yang ditanamkan dalam mesin server, memiliki peran untuk pengelolaan transaksi data secara otomatis, sehingga ketika terdapat permintaan atau perintah dari operator, maka *server* akan langsung merespon dan menindaklanjuti untuk mengirimkan perintah pada IoT-*Client*. Konsep M2M dalam IoT mewajibkan alat atau sistem dapat mengenali secara langsung secara otomatis format data yang dilewatkan dalam protokol TCP/IP. REST-API pada penelitian terdahulu digunakan sebagai jembatan tampilan data antara sistem berbasis website dan

sistem *mobile* [14]. REST-API pada sistem IoT bertindak sebagai bentuk arsitektur komunikasi REST (*REpresentational State Transfer*) dengan sifat *Stateless* informasi. Informasi permintaan klien (operator) melalui jendela tampilan antar muka website disajikan dalam Form-HTML. Ketika operator melakukan perubahan parameter, data akan dikirim menuju *server* dan dilakukan pemutakhiran terkait konfigurasi informasi perangkat IoT-*Client* yang disimpan dalam basis data. Setelah dilakukan pemutakhiran informasi perangkat IoT-*Client* pada sisi *server* kemudian *server* akan mengkonversi dan mengirimkan informasi dalam bentuk format data JSON dari sisi mesin server menuju pada perangkat IoT-*Client*. Data JSON yang dikirim nantinya akan diterjemahkan oleh pihak IoT-*Client* untuk menentukan *task* (tugas) yang harus dikerjakan oleh perangkat tersebut.

REST-API untuk menangani data, kendali dan permintaan perekaman basis data perangkat IoT-*Client* disajikan dalam Tabel 1, dalam tabel tersebut terdapat empat macam perintah dasar yang nantinya berfungsi sebagai pemantauan kondisi alat, pemutahiran data alat, pengiriman JSON dan penyimpanan data. Perintah-perintah tersebut akan secara terus menerus berjalan dalam sisi *server*. Ketika terdapat perubahan data atau kondisi, maka REST-API yang terintegrasi dalam mesin server akan memberikan *feedback* kepada seluruh *client* (baik *Client* yang berupa instrumen IoT dan *Client* manusia yang mengoperasikan komputer). Pada REST-API yang dibuat dalam mesin server, terdapat penggunaan metode GET untuk proses insert data dalam basis data. Penggunaan method GET dengan parametrik URI berupa segmen data digunakan untuk melakukan injeksi data dari IoT-*Client* (perangkat RPI-Pico W) menuju mesin basis data. Metode GET yang digunakan sebagai penambahan data, hanya boleh dipanggil oleh klien yang bersifat perangkat keras (perangkat IoT-*Client*) dan bukan oleh klien yang bersifat operator. Alat akan membangkitkan kode unik yang digunakan sebagai penanda bahwa peng-*input* data adalah mesin dan bukan manusia. Saat dilakukan insert data, perintah GET tidak akan tampil dalam layar karena hal tersebut diproses secara langsung dalam perangkat *embedded system* IoT.

Tabel 1. REST-API pada Server IoT

No	Nama Layanan	Metode	Path/URI	Luaran/Peran
1	Tampilkan status alat	GET	/kendali	Kondisi alat dalam tabel relasi tampilan website
2	Proses update status alat	POST	/proses	Kirim data isian form dari komputer *Client* menuju server
3	Kirim data JSON menuju IoT-*Client*	GET	/jsondata	Data JSON yang dapat dibaca oleh alat IoT-*Client*
4	Query penyimpanan data dalam mesin basis data MySQL	GET	/tambah/(:segmen1-n)	Penyimpaan data dalam mesin basis data

Sistem kerja komutasi dalam sisi instrumen, micropython yang mana memiliki domain pemrograman python akan memantau secara berulang dan terus menerus sejak instrumen tersebut dinyalakan sampai dengan instrumen tersebut mati. Pemrograman micropython memiliki fungsi yang analog dengan sebuah perangkat sistem operasi yang terus *standby* dan *memonitoring* perubahan-perubahan data. Adapun *pseudocode* dalam instrumen RPi Pico W untuk menjalankan fungsi ganda dari perangkat IoT-*Client* yaitu sebagai kendali aktuator dan basis layanan ditunjukkan dalam kode 1. Pada potongan *pseudocode* tersebut, diawali dengan pemanggilan pustaka untuk pengaturan pada sistem jaringan internet. Kemudian dilakukan pengaturan perangkat IOT sebagai *client* atau *server*. Pada penelitian ini RPi Pico W difungsikan sebagai unit IoT-*Client*. Setelah RPi Pico W terhubung dalam jaringan, alat tersebut akan menginjeksikan perintah cek status yang nantinya akan direspon oleh *server*. Melalui REST-API data JSON akan dikirimkan kepada alat atau instrumen. Kemudian instrumen RPi Pico W akan menerjemahkan data JSON melalui perintah ekstraksi informasi JSON. Seluruh penerjemahan aset informasi yang terpaket dalam JSON, nantinya akan digunakan oleh IoT-*Client* untuk tindakan pengambilan keputusan (eksekutor) terhadap data permintaan klien (operator) terhadap kondisi aktuator apakah LED menyala atau padam dan melayani server dalam penyediaan data yang nantinya akan disimpan dalam basis data.

Kode 1. Micropyhon dalam Raspberry Pi Pico W (IoT *Client*)

```
Library tambahan:
import network-import urequests-import json
from machine import Pin-from machine import Timer

Koneksi Jaringan sebagai Client:
wlan = network.WLAN(network.STA_IF)
wlan.active(True)

Injeksi perintah pada server:
urequests.get("path URL/URI"

Ekstraksi JSON:
data_json=r.json()
json.loads(data_text)[0].get('status_a')

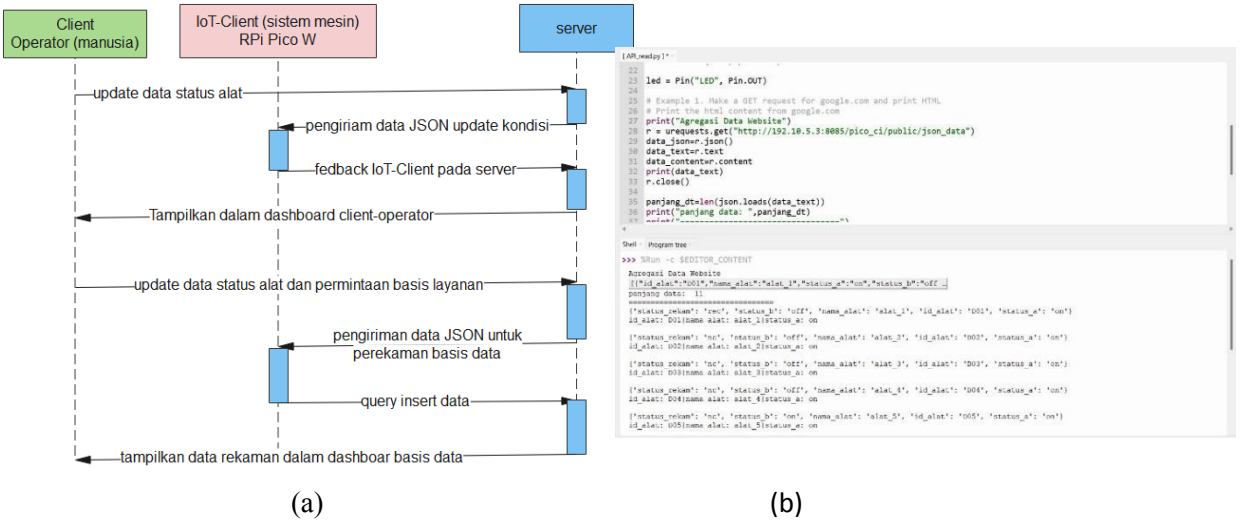
pengambilan putusan
jika terdeteksi sebagai Client-Eksekutor (pelaksana
pengondisian led ON/OFF (Pin("LED", Pin.OUT)=>on/off)
jika terdeteksi sebagai Client-basis pelayanan => perekaman
data dengan fungsi:
urequests.get("http://domain server/tambah/(segmen1-n)")
```

Konsep IoT-*Client* menggunakan RPi Pico W dapat dianalogikan unit tersebut sebagai seseorang yang bertugas sebagai pengambilan putusan atas kondisi informasi dan sekaligus pelayan untuk kegiatan *input* data secara otomatis. Peran fungsi ganda tersebut membutuhkan struktur kerangka kerja yang saling sinkron antara sisi *client* dan *server*. Pada penelitian ini sistem sinkronisasi dibatasi dalam kerangka komunikasi data sinkron, oleh sebab itu pemrosesan data dalam komputasi bersifat hierarki merujuk pada algoritma. Secara umum sistem yang dirancang memiliki alur pemrosesan data yang disajikan dalam Gambar 3.a dengan bentuk diagram berjenjang merepresentasikan seluruh proses dilakukan secara berurutan. Dalam sistem IoT yang telah dirancang dapat dibagi menjadi dua jenis *client*, yaitu *client* manusia atau operator dan *Client* berbasis mesin. *Client* manusia mengoperasikan komputer melalui web-browser untuk mengakses informasi dan interaktif dengan unit *server*. *Server* yang merupakan murni mesin komputer, akan memberikan pelayanan berdasarkan permintaan dan jenis *client*. Pada *client* yang bersifat operator, server akan memberikan *feedback* berupa update data pada tampilan dashboard. Sedangkan pada IoT *Client* – berbasis mesin, server akan memberikan *feedback* berupa data JSON yang nantinya diterjemahkan oleh sistem micropython.

Micropython mengekstraksi untaian teks JSON hingga menjadi informasi spesifik. Pengambilan dalam bentuk larik dilakukan dengan merujuk pasangan kunci dan nilai (*key:value*). Data kunci akan dicari dengan proses algoritma pencarian. Proses pencarian nilai akan dilakukan berulang dengan merujuk pada banyaknya perangkat. Proses ekstraksi data yang telah ditemukan, nantinya digunakan oleh mesin IoT-*Client* RPi untuk menentukan tindakan berikutnya. Aksi yang dilakukan RPi Pico W sebagai unit eksekutor adalah menyalakan perangkat elektronik (dalam kasus penelitian ini digunakan lampu LED). Apabila *Client* operator meminta server untuk sisi *Client*-RPi sebagai unit layanan, maka perangkat RPi akan memberikan layanan untuk mengirimkan data pada mesin basis data. Pengiriman data oleh RPi dilakukan secara otomatis dengan merujuk pada query insert MySQL.

2.5 Tahap Kelima - Pengujian Sistem IoT

Rumusan IoT-*Client* pada domain RPi Pico W dengan dual fungsional yaitu sebagai unit eksekutor dan basis pelayanan diperlukan pengujian bertahap. Pengujian tersebut meliputi aspek pengujian kesesuaian tampilan terhadap isi status data instrumen IoT, pengujian waktu respon perekaman basis data, kesesuaian nilai data dengan aktuator perangkat keras dan validasi data sensor yang terekam dalam mesin basis data.



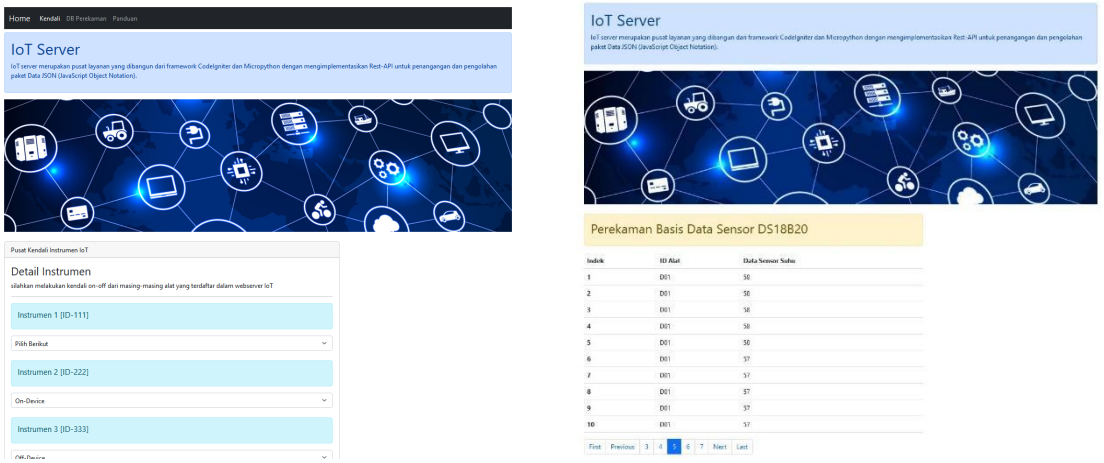
Gambar 3 a) Diagram berjenjang proses komunikasi data dalam *Client*-Server IoT, b) format data JSON yang terbaca oleh RPi sebagai unit *Client*-IoT.

3. Hasil dan Pembahasan

Merujuk dari penelitian yang telah dilakukan, terdapat hasil lauran berupa aplikasi antar muka, desain REST-API komunikasi *back-end* IoT, dan pengukuran parameter dari aktivitas komunikasi *client*-server IoT berbasis Raspberry Pi Pico W, adapun paparan pengujian adalah sebagai berikut:

3.1. Luaran Penyusunan Aplikasi Antar Muka IoT

Seperti pada keterangan sebelumnya dua macam klien yaitu IoT-Client dan operator menjadikan dalam sistem tersebut dibuat dua macam antar muka. Tampilan berbasis web-GUI merupakan bentuk antarmuka operator dengan sisi server. Aplikasi untuk pengaturan IoT-*Client* RPi dirancang seperti pada Gambar 4, dalam rancangan tersebut terdapat fitur pemilihan kondisi untuk proses menyalakan dan mematikan dari aktuator. *Form*-HTML yang terpasang dalam antarmuka tersebut, nantinya akan mengirimkan data dengan metode POST untuk pemutakhiran basis data khususnya pada tabel yang memuat informasi kondisi status instrumen Raspberry Pi Pico W yang bertindak sebagai IoT-*Client*. Perintah yang dikirim melalui form berisikan status pengaturan aktuator LED. Instrumen aktuator yang dipasang dalam sistem adalah lampu LED yang nantinya akan menyala atau mati ketika pengguna melakukan perubahan status. Pada sistem antar muka, disajikan banyak instrumen yang tergabung dalam satu jaringan IoT. Pada penelitian ini secara basis data, instrumen memiliki ID-Alat unik sehingga akan memudahkan dalam identifikasi informasi. Apabila pengguna mengatur IoT-Client sebagai unit kendali LED, maka aktuator akan merespon dengan kondisi yang ada dan apabila pengguna mengatur IoT-Client sebagai unit layanan untuk pengiriman data (perekaman basis data) suhu sensor DS18B20, maka server akan menanggapi dengan melakukan perintah *insert* data.



(a) (b)

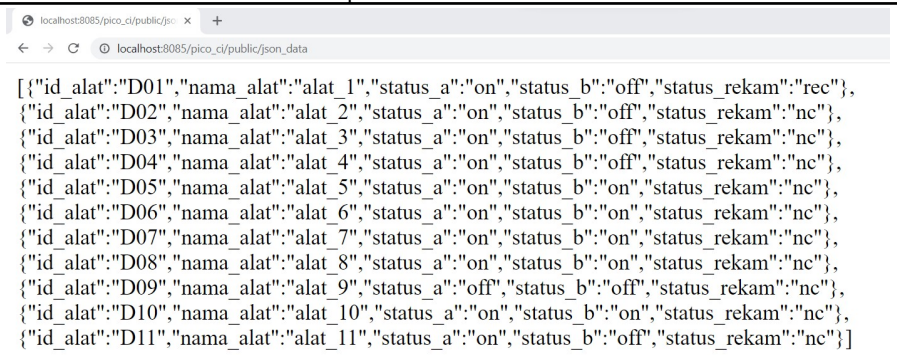
Gambar 4. a) Tampilan antar muka untuk menjalankan aktuator berupa nyala-mati lampu LED, b) tampilan daftar basis data ketika dilakukan perekaman oleh sensor suhu.

3.2. Validasi Data JSON dan Kesesuaian Instrumen IoT-*Client*

Data JSON yang dikirim antar *client*-Server dalam sistem IoT memiliki format seperti pada Gambar 5, dalam teks tersebut terdapat pasangan *key-value* yang merepresentasikan kondisi dari instrumen dan nilai yang menjadi acuan oleh RPi untuk dilakukan aksi. Nilai tersebut dapat dimutakhirkan oleh pengguna melalui akses c*lient*-operator dengan mengganti beberapa parameter dari tampilan antar muka yang telah didesain. Pada JSON tersebut terdapat empat parameter yang terhubung dengan nilai. Pada Tabel 2 disajikan penjelasan makna dari masing-masing nilai yang berelasi dari kunci.

Tabel 2. Pasangan Key-Value dan peran dalam komunikasi IoT

No	Key	Value	Peran/Fungsional
1	id_alat	D01	Kode unik ID alat agar data tidak tidak tertukar dan dapat dengan mudah diidentifikasi sistem IoT-RPi
2	nama_alat	Alat_1	Penamaan alat, nilai tersebut hanya bersifat deksripsi singkat untuk mempermudah identifikasi sistem
3	status_a	On	Pernyataan untuk pengambilan putusan nyala-mati dari lampu LED dari kanal A
4	status_b	Off	Pernyataan untuk pengambilan putusan nyala-mati dari lampu LED dari kanal B
5	status_rekam	rec	Pernyataan untuk memposisikan isntrumen *Client* IoT apakah sebagai unit perekaman (layanan) atau hanya bersifat eksekutor untuk menyalakan lampu LED



Gambar 5. Data JSON sebagai representasi status instrumen dan perintah *Client* IoT.

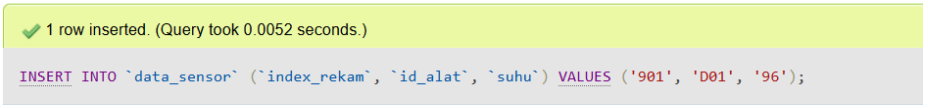
Tabel 3. Pasangan Key-Value dan peran dalam komunikasi IoT

Pengujian ke	perintah	Kondisi LED	Interval penekanan (detik)
1 - 23	Status_a =”on”	Menyala	2
24; 25; 26; 34; 35;36	Status_a =”on”	Padam	0.5
27 - 33	Status_a =”on”	Menyala	1
37 - 50	Status_a =”on”	Menyala	1

Pengujian kesesuain antara perintah dan kondisi LED disajikan dalam Tabel 3, pengujian on-off pada kontrol nyala LED dilakukan dengan variasi waktu respon. Sistem IoT-Client Raspberry Pi melalui jembatan komunikasi client-server dapat merespon perubahan kondisi logic 1-0 dengan interval skala waktu minimal 1 detik. Total pengujian yang telah dilakukan yaitu sekitar 50 kali proses on-off lampu LED didapatkan hasil 88% data sesuai antara kondisi JSON dan logika on-off dari lampu sebagai aktuator. Proses kesalahan atau kegagalan (ketidaksesuaian data JSON dan kondisi lampu) disebabkan karena *overload* data jaringan jika perintah dikirimkan secara berut kurang dari 1 detik. Pada saat perekaman basis data bersamaan dengan aktivitas menyalakan aktuator akan terjadi error karena ada *overlap* data dalam komunikasi TCP/IP. Hal tersebut menyebabkan perlu waktu jeda antar perintah.

3.3. Pengujian Waktu Respon Perekaman Data

Pengujian respon perekaman data dalam mesin basis data, didapatkan nilai sekitar 0.0052 detik. Merujuk pada nilai tersebut, mesin MySQL dapat dengan mudah untuk melakukan perekaman basis data pada tiap satu detik. Sistem IoT-*Client* RPi memiliki jeda waktu pengiriman data yaitu 2 detik tiap pengiriman data. Pada Gambar 6 merupakan waktu komputasi untuk pemrosesan satu kali perekaman data dalam mesin basis data MySQL. Adanya waktu yang cukup singkat dari proses perekaman tersebut dipengaruhi oleh pembebanan sistem ketika terjadi dua aktivitas berjalan (perekaman basis data dan on-off aktuator).



Gambar 6. Waktu komputasi perekaman satu data dalam mesin basis data MySQL

3.4. Validasi Transmisi Data Sensor

Perekaman data sensor yang terkirim pada mesin basis data, memiliki nilai deviasi yang cukup kecil. Proses pengukuran dilakukan dengan melakukan komparasi dengan alat ukur standar yaitu thermometer digital. Hasil pembacaan dengan merujuk pada 30 data pengukuran, didapatkan nilai deviasi sekitar 0.1 ^{0}C. Pengambilan data pada DS18B20 memiliki nilai luaran decimal, akan tetapi saat dilakukan transmisi data pada MySQL dilakukan konversi data menjadi data bilangan bulat (integer).

4. Kesimpulan

Merujuk dari serangkaian penelitian yang telah dilaksanakan, dapat ditarik kesimpulan sebagai berikut: fungsi ganda IoT-*Client* berbasis Raspberry Pi Pico W dapat dilakukan dengan baik untuk aktivasi aktuator dan basis layanan dalam perekaman basis data; integrasi Micropython dan REST-API dengan basis pemrograman PHP-CodeIgniter dapat menjadi alternatif manajemen pertukaran data IoT sebagai alternatif arsitektur MQTT-Broker; Proses pembebanan sistem dalam fungsi ganda IoT-Client Raspberry Pi Pico W memberikan kontribusi validasi data kendali aktuator sekitar 88% dan waktu jeda sekitar 1-2 detik; hasil pengukuran waktu komputasi dari perekaman basis data didapatkan nilai sekitar 0.0052.

Referensi

[1] V. S. Chakravarthi, *Internet of Things and M2M Communication Technologies: Architecture and Practical Design Approach to IoT in Industry 4.0*. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-79272-5.

[2] C. Anton-Haro dan M. Dohler, Ed., *Machine-to-machine (M2M) communications: architecture, performance and applications*. dalam Woodhead Publishing series in electronic and optical materials, no. number 69. Cambridge, UK: Woodhead Publishing Limited, 2015.

[3] L. M L dan A. M, "LoRa technology for Internet of Things(IoT):A brief Survey," dalam *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India: IEEE, Okt 2020, hlm. 8–13. doi: 10.1109/I-SMAC49090.2020.9243449.

[4] B. Padma dan S. B. Erukala, "End-to-end communication protocol in IoT-enabled ZigBee network: Investigation and performance analysis," *Internet of Things*, vol. 22, hlm. 100796, Jul 2023, doi: 10.1016/j.iot.2023.100796.

[5] C. Gupta dan G. Varshney, "An improved authentication scheme for BLE devices with no I/O capabilities," *Computer Communications*, vol. 200, hlm. 42–53, Feb 2023, doi: 10.1016/j.comcom.2023.01.001.

[6] S. Aheleroff *dkk.*, "IoT-enabled smart appliances under industry 4.0: A case study," *Advanced Engineering Informatics*, vol. 43, hlm. 101043, Jan 2020, doi: 10.1016/j.aei.2020.101043.

[7] A. S. Tanenbaum dan D. Wetherall, *Computer networks*, Fifth edition, Pearson new international edition. Place of publication not identified: Pearson India Education Services Pvt, Limited, 2018.

[8] E. Longo dan A. E. C. Redondi, "Design and implementation of an advanced MQTT broker for distributed pub/sub scenarios," *Computer Networks*, vol. 224, hlm. 109601, Apr 2023, doi: 10.1016/j.comnet.2023.109601.

[9] S. Yousefi, H. Karimipour, dan F. Derakhshan, "Data Aggregation Mechanisms on the Internet of Things: A Systematic Literature Review," *Internet of Things*, vol. 15, hlm. 100427, Sep 2021, doi: 10.1016/j.iot.2021.100427.

[10] O. Safitri, T. Andriani, P. A. Topan, dan I. Darmawan, "Implementasi metode fuzzy logic pada rancang bangun bak sampah buka tutup otomatis berbasis raspberry pi pico," *Journal Altron; Journal of Electronics, Science & Energy systems*, vol. 2, no. 01, Art. no. 01, Feb 2023.

[11] A. Weisrock, V. Couty, J.-F. Witz, L. Thorrez, dan P. Lecomte-Grosbras, "CRAPPY goes embedded: Including low-cost hardware in experimental setups," *SoftwareX*, vol. 22, hlm. 101348, Mei 2023, doi: 10.1016/j.softx.2023.101348.

[12] D. Dinculeană dan X. Cheng, "Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices," *Applied Sciences*, vol. 9, no. 5, Art. no. 5, Jan 2019, doi: 10.3390/app9050848.

[13] A. Elyounsi dan A. N. Kalashnikov, "Evaluating Suitability of a DS18B20 Temperature Sensor for Use in an Accurate Air Temperature Distribution Measurement Network," *Engineering Proceedings*, vol. 10, no. 1, Art. no. 1, 2021, doi: 10.3390/ecsa-8-11277.

[14] R. Ramadhan dan P. Purwanto, "Implementasi Web Service Rest API Untuk Merancang Aplikasi Pusat Informasi Masjid Al Muhajirin Larangan Indah," *Prosiding Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, vol. 1, no. 1, Art. no. 1, Sep 2022.

How to cite this article:

Albanna I, Hendro N. Fungsi Ganda Perangkat IoT-Client Sebagai Kendali Aktuator dan Basis Layanan Melalui Komunikasi Paket Data JSON. *Jurnal Teknologi dan Manajemen.* 2023 Juli; 4(2):119-129. DOI: 10.31284/j.jtm.2023.v4i2.4699