Sistem Kendali Terpusat Penjadwalan Perangkat *Air Conditioner* Berbasis *Internet of Things*

http://dx.doi.org/10.28932/jutisi.7i2.3448

Riwayat Artikel
Received: 24 Februari 2021 | *Final Revision*: 15 Juli 2021 | *Accepted*: 23 Juli 2021

Daniel Felix Setiawan[#1], Laurentius Kuncoro Probo Saputra[✉*2], Yuan Lukito[#3]

Informatika, Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana
Jl. Dr. Wahidin Sudirohusodo no. 5-25, Yogyakarta

[1]daniel.felix@ti.ukdw.ac.id
[2]kuncoro@staff.ukdw.ac.id
[3]yuanlukito@ti.ukdw.ac.id

Abstract — **Air Conditioner (AC) devices in Duta Wacana Christian University 2nd floor Agape computer laboratory are turned on in the morning and turned off in the evening manually by the computer laboratory officer every day after all the classes are over. Whereas there are sessions when laboratory rooms are not used, but the AC is still on. So, there is a waste of electricity. It wastes time and energy in vain. Therefore, the implementation of AC control and scheduling can be the solution. The implementation includes a program for PC that is used to control the AC and IR module that is attached to the computer laboratory's AC. IR module consists of NodeMCU ESP8266 microcontroller and IR LED. All microcontrollers are connected to the internet over a WiFi network. The microcontroller substitutes conventional remote control. From the result, it can be concluded that the AC control and scheduling system is successfully implemented and worked well with good performance. After the system is implemented, computer laboratory officers no longer need to turn on and off the AC manually every day. Therefore, it can save energy and time. Besides that, waste of electrical energy can be minimized, because AC will turn off automatically when the laboratory room is not used.**

Keywords— **air conditioner control system; air conditioner scheduling device; infrared wireless control module; internet of things.**

I. PENDAHULUAN

Pengoperasian perangkat pendingin ruangan (AC) di area laboratorium komputer Agape lantai 2 Universitas Kristen Duta Wacana (UKDW) memerlukan petugas yang harus berkeliling ruangan laboratorium untuk menghidupkan dan mematikan AC secara manual satu per satu setiap harinya. Hal tersebut sangatlah tidak efektif karena membuang waktu dan energi. Selain itu, AC di laboratorium komputer Agape lantai 2 UKDW dihidupkan pada pagi hari dan dimatikan pada malam hari seusai sesi terakhir perkuliahan. Padahal, ada beberapa sesi di mana ruang laboratorium tidak digunakan atau kosong, tetapi AC di ruang laboratorium yang kosong tersebut tetap menyala. Di sini jelas terjadi pemborosan listrik.

Implementasi sistem kendali terpusat serta penjadwalan untuk AC di setiap ruang laboratorium komputer Agape lantai 2 Universitas Kristen Duta Wacana dapat membuat pekerjaan menjadi lebih efektif, efisien, dan hemat listrik. Dengan diimplementasikannya sistem ini, pemborosan listrik dapat diminimalkan, karena AC hanya akan beroperasi sesuai jadwal yang telah diatur. Dengan kata lain, AC akan dimatikan secara otomatis jika ruang tidak digunakan. Selain itu, juga sudah tidak memerlukan petugas yang harus berkeliling laboratorium untuk menghidupkan AC di pagi hari dan mematikan AC pada malam hari, karena sistem ini yang akan melakukan pekerjaan tersebut secara otomatis sesuai jadwal.

Menurut Junaidi [1], penggunaan komputer saat ini mampu mendominasi pekerjaan manusia seperti mengontrol peralatan elektronik dari jarak jauh menggunakan Internet. Konsep jaringan *Internet of Things* (IoT) memungkinkan pengguna untuk mengelola dan mengoptimalkan penggunaan berbagai macam peralatan elektronik menggunakan Internet. Dengan adanya IoT, diharapkan benda-benda fisik akan dilengkapi dengan berbagai jenis sensor dan terhubung ke Internet melalui jaringan serta dukungan teknologi, seperti pada implementasi IoT untuk pengendalian lampu [2] dan dapat pula dikembangkan dengan menerapkan *artificial intelligence* seperti penelitian Hakim dkk.[3]. Penerapan yang sangat fleksibel membuat IoT banyak diimplementasikan juga di dunia industri, dimana alat-alat seperti SCADA dapat juga dikendalikan jarak jauh [4].

Semua AC yang ada di Universitas Kristen Duta Wacana masih berupa AC konvensional (belum *smart appliance*) tetapi sudah dilengkapi dengan sensor penangkap inframerah, sehingga dapat dikontrol menggunakan *remote control* berbasis inframerah. Dengan teknologi *microcontroller* yang ada saat ini, konsep dan cara kerja *remote control* tersebut dapat ditiru dan digantikan fungsinya oleh *microcontroller* seperti Arduino, Node MCU, Raspberry Pi, dan lain-lain. *Microcontroller* juga dapat terhubung satu sama lain melalui jaringan komputer, baik menggunakan kabel ataupun wireless seperti konsep jaringan IoT. Dengan konsep tersebut, sistem kendali terpusat dan penjadwalan AC dapat dibuat dengan memanfaatkan rangkaian *microcontroller* sebagai *remote control* yang dapat mengontrol AC sesuai jadwal yang telah ditentukan pada program pengguna.

Laurentius Kuncoro Probo Saputra dan Yuan Lukito [5] dalam penelitiannya membuat suatu sistem kontrol AC menggunakan protokol RESTful dan *microcontroller* NodeMCU ESP8266. Dalam penelitiannya, perangkat inframerah (IR) WiFi module yang dipasangkan pada perangkat AC. Modul tersebut terdiri dari *microcontroller* NodeMCU ESP8266, IR LED, dan sensor LDR (Light Dependent Resistor). Modul tersebut digunakan untuk mengupgrade AC split konvensional yang ada di gedung Universitas Kristen Duta Wacana agar dapat dikontrol melalui smartphone atau PC yang terhubung ke jaringan WiFi dan dapat dicek kondisinya (hidup/mati) dari jarak jauh. Di akhir penelitian ini terbukti bahwa IR WiFi module dapat digunakan untuk mengontrol AC menggunakan smartphone atau laptop via browser. Parameter pengukuran dalam penelitiannya menggunakan data yang berisi *connection time*, *waiting time*, dan *receiving time*. *Connection time* adalah waktu yang diperlukan *browser* untuk terhubung ke NodeMCU. *Waiting time* adalah waktu yang diperlukan NodeMCU untuk melakukan *parsing* terhadap perintah yang dikirimkan, mengirim sinyal inframerah ke AC, dan mengirimkan respon dalam bentuk JSON. Sedangkan *receiving time* adalah waktu yang diperlukan untuk mengirim JSON data dari NodeMCU ke *browser*. Hasil penelitiannya mendapatkan *connection time* tercepat adalah 7ms. *Connection time* yang diperoleh tidak stabil. *Waiting time* rata-rata adalah sebesar 527,85 ms. *Waiting time* yang diperoleh cukup stabil. *Receiving time* yang diperoleh tidak stabil dengan rata-rata sebesar 65,92 ms.

Erham, Markus, dan Sopianti [6] dalam penelitiannya membuat suatu sistem yang dapat menghidupkan dan mematikan AC *window* secara otomatis berdasarkan suhu ruangan. Sistem menerima masukan dari pengguna berupa *set point temperature* dan *differential value*. Selain itu, sistem juga menerima masukan dari sensor suhu SHT 1x berupa informasi suhu ruangan. Kemudian sistem akan menghitung nilai *threshold* $T_{cut\text{-}off}$ dan $T_{cut\text{-}in}$. Setelah mendapatkan nilai *threshold* $T_{cut\text{-}off}$ dan $T_{cut\text{-}in}$, sistem akan membandingkan suhu ruangan dengan kedua nilai *threshold* tersebut. Jika suhu ruangan $\leq T_{cut\text{-}off}$ (terlalu dingin), maka sistem akan memerintahkan relay untuk memutus arus yang terhubung ke AC, sehingga AC akan mati. Jika suhu ruangan $\geq T_{cut\text{-}in}$ (terlalu panas), maka sistem akan memerintahkan relay untuk menyambung arus yang terhubung ke AC, sehingga AC akan hidup. Hal ini akan diulang secara terus-menerus untuk menjaga suhu ruangan agar tetap berada pada suhu yang diinginkan / *set point temperature*. Di akhir penelitian ini dilakukan pengambilan data untuk dianalisis. Pengambilan data dilakukan untuk 3 *set point temperature* yang berbeda, yaitu 22°C, 23°C, dan 24°C dengan *differential value* yang sama, yaitu sebesar 2°C. Berdasarkan data yang didapatkan, setiap kurva mengalami pergerakan naik turun di sekitar *setpoint temperature*. Kurva dengan *setpoint temperature* 22°C mengalami siklus on-off paling sering karena waktu yang diperlukan dari $T_{cut\text{-}off}$ untuk mencapai $T_{cut\text{-}in}$ pada kurva *setpoint temperature* 22°C adalah paling singkat. Tetapi, frekuensi on/off untuk ketiga *setpoint temperature* tersebut tidak berbeda secara signifikan. Sehingga, dapat disimpulkan bahwa perbedaan *setpoint temperature* tidak memengaruhi konsumsi energi listrik secara signifikan.

Ramesh, Kumar, Vamsi, dan Akarsh [7] dalam penelitiannya membuat suatu sistem *remote* universal dalam bentuk aplikasi *mobile smartphone* yang dapat mengontrol perangkat AC melalui jaringan Wi-Fi. *Microcontroller* yang digunakan adalah NodeMCU ESP8266. Langkah pertama dalam penelitian tersebut adalah merekam data-data sinyal dari beberapa *remote control* asli yang diperlukan menggunakan *IR library* (*library* inframerah untuk Arduino). Perekaman sinyal dilakukan dengan membuat program yang digunakan untuk merekam sinyal ke NodeMCU ESP8266, kemudian menekan tombol pada *remote control* dengan posisi *remote control* diarahkan ke penerima inframerah. Ketika sebuah tombol pada *remote control* ditekan, maka *remote control* akan mengeluarkan sinyal inframerah. Data sinyal inframerah inilah yang direkam menggunakan *IR library* Arduino. Data-data sinyal ini kemudian digunakan untuk mengontrol berbagai peralatan elektronik menggunakan *microcontroller* NodeMCU ESP8266. Langkah selanjutnya adalah membuat program untuk dimasukkan ke NodeMCU ESP8266. NodeMCU tersebut akan berperan sebagai *web server* yang dapat diakses melalui *IP address*-nya. Ketika diakses, akan muncul sebuah halaman yang berisikan tombol-tombol dengan berbagai perintah untuk mengendalikan perangkat elektronik. Ketika pengguna menekan tombol yang ada pada halaman tersebut, *microcontroller* akan memproses dan mengeluarkan sinyal ke IR LED sesuai perintah yang diberikan oleh pengguna.

Berdasarkan permasalahan yang dihadapi dan tinjauan pustaka yang telah dilakukan maka tujuan yang ingin dicapai pada penelitian ini adalah mengimplementasikan sistem kendali terpusat dan penjadwalan terhadap AC di laboratorium komputer Agape lantai 2 Universitas Kristen Duta Wacana dalam bentuk aplikasi desktop yang dapat berkomunikasi dengan seluruh perangkat IR wireless modul yang terpasang pada AC menggunakan protokol *Message Queuing Telemetry Transport* (MQTT).

Manfaat yang didapatkan dari penelitian ini adalah dapat meminimalkan pemborosan energi listrik di Universitas Kristen Duta Wacana dan mengurangi pekerjaan para petugas untuk menghidupkan dan mematikan AC setiap harinya, sehingga waktunya bisa dialokasikan untuk melakukan pekerjaan lain.

II. METODE

A. Pengumpulan Data

Tahap ini merupakan tahap pengumpulan data yang berkaitan dengan penelitian ini, seperti daftar merek AC yang digunakan di laboratorium komputer Agape lantai 2 UKDW, protokol pensinyalan yang digunakan untuk mengontrol AC, dan jadwal pemakaian ruang laboratorium. Survey dilakukan pada setiap ruangan laboratorium komputer Agape lantai 2 UKDW untuk melihat merek AC yang digunakan di setiap ruang. Hampir seluruh perangkat AC dari berbagai merek difasilitasi dengan perangkat *remote control* berbasis inframerah.

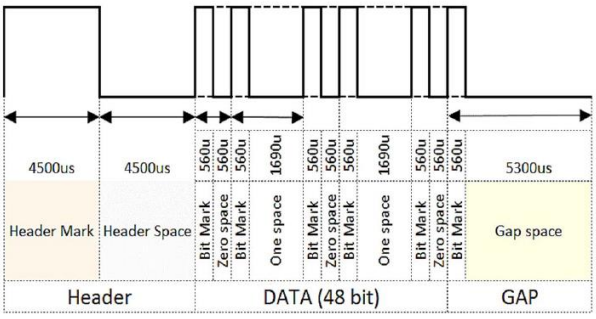
Inframerah merupakan gelombang elektromagnetik yang memiliki frekuensi di bawah spektrum cahaya tampak, sehingga tidak terlihat oleh mata manusia, namun dapat terdeteksi oleh *receiver* inframerah yang ada pada peralatan elektronik. Pada transmisi inframerah terdapat 2 terminologi yang sangat penting, yaitu *space* yang menyatakan tidak ada sinyal *carrier* dan *pulse* yang menyatakan adanya sinyal *carrier*.

Menurut [8], ketika sebuah tombol pada *remote control* ditekan, maka *transmitter* inframerah akan mengeluarkan suatu sinyal dengan frekuensi 30 – 40 kHz dalam bentuk sinyal kotak ter-*encode* yang akan dideteksi sebagai urutan data biner. Bentuk kode sinyal untuk masing-masing tombol tergantung pada perusahaan produsen peralatan elektronik yang membuatnya. Pada dasarnya setiap perusahaan bebas menentukan kode sinyal untuk setiap tombol pada *remote control*. Penggunaan sinyal inframerah ini hanya cocok untuk di dalam ruang saja, seperti pada peralatan elektronik rumah atau kantor, karena memiliki keterbatasan jarak yang pendek (maksimal sekitar 10 meter) dan sudut pengiriman juga kecil. Sehingga *remote control* harus diarahkan tepat ke alat elektronik tersebut. Sinyal inframerah juga tidak bisa menembus dinding, sehingga antara *transmitter* dan *receiver* harus berada di 1 ruang yang sama.

Beberapa sumber cahaya seperti matahari, bola lampu, lilin, dan lain-lain juga memancarkan cahaya inframerah. Oleh karena itu, harus diambil beberapa tindakan pencegahan untuk menjamin bahwa sinyal inframerah yang akan dikirim ke penerima tidak mengalami interferensi. Tindakan pencegahan dilakukan dengan cara memodulasi data yang dikirimkan pada frekuensi tertentu. Penggunaan modulasi ini dimaksudkan untuk mencegah kesalahan penerima (*receiver*) dalam menerjemahkan data dari pengirim (*transmitter*) dan membuat sinyal menjadi lebih kebal terhadap *noise*/interferensi. Modulasi akan membuat LED inframerah berkedip pada frekuensi tertentu. Penerima inframerah juga akan diatur pada frekuensi tersebut, sehingga semua cahaya inframerah dengan frekuensi yang berbeda akan diabaikan.

Sebuah sistem *remote control* terdiri dari beberapa bagian:

1) *Transmitter (pengirim sinyal):* Alat ini berfungsi untuk mengirimkan suatu instruksi ke peralatan elektronik yang ingin dikontrol. Alat ini berupa sebuah LED inframerah yang berada di pesawat *remote control*. LED inframerah adalah sejenis lampu kecil berupa dioda yang akan memancarkan cahaya inframerah apabila diberi arus.
2) *Panel remote control:* Panel ini berisi sejumlah tombol di pesawat *remote control*. Setiap tombol memiliki fungsi yang spesifik.
3) *Papan rangkaian elektronik:* Di dalam setiap pesawat *remote control* terdapat sebuah papan rangkaian elektronik atau PCB. Di dalam PCB tersebut terdapat IC (Integrated Circuit) yang berfungsi untuk mendeteksi penekanan tombol pada *remote control* tersebut, kemudian membangkitkan transmitter untuk mengirimkan sinyal sesuai tombol yang ditekan
4) *Receiver (penerima sinyal):* Alat ini berada di dalam alat elektronik yang akan menerima instruksi. Alat ini berupa foto transistor inframerah yang berperan dalam mendeteksi sinyal inframerah yang dikirimkan *remote control*.



Gambar 1. Contoh Spesifikasi Protokol Inframerah [5]

Merek AC dibutuhkan untuk menentukan protokol inframerah yang digunakan untuk masing-masing merek. Protokol inframerah remote AC yang perlu diperhatikan ialah lebar pulsa waktu yang digunakan sebagai penanda untuk setiap bit *header mark, header space, bit mark, zero space, one space,* dan *gap space. Header mark* dan *space* digunakan sebagai penanda awal perintah signal, *bit mark* merupakan penanda bahwa data berikutnya ialah data bit signal, *zero space* merupakan penanda bit 0, dan *one space* merupakan penanda bit 1. Setiap bit ditransmisikan menggunakan lebar pulsa yang berbeda-beda mengikuti jenis bit dan protokol yang digunakan, seperti pada Gambar 1.

Proses pengumpulan data-data spesifikasi lebar pulsa untuk setiap merek AC dilakukan dengan merekam sinyal dari masing-masing *remote control* asli menggunakan *IR library* (*library* inframerah untuk Arduino). Perangkat keras yang dibutuhkan untuk merekam sinyal adalah 1 buah *microcontroller* yang terhubung ke *receiver* inframerah. Perekaman sinyal dilakukan dengan meng*upload* program yang digunakan untuk merekam sinyal ke *microcontroller*, misalnya Arduino Uno, NodeMCU ESP8266, atau yang lainnya, kemudian menekan tombol pada *remote control* dengan posisi *remote control* diarahkan ke *receiver* inframerah. Ketika suatu tombol pada *remote control* ditekan, maka *remote control* akan mengeluarkan sinyal inframerah. Data sinyal inframerah inilah yang akan direkam menggunakan *IR library* Arduino. Sinyal yang akan direkam adalah sinyal untuk menghidupkan dan mematikan AC, sinyal untuk setiap level suhu yang didukung oleh masing-masing AC (misalnya dari 16 sampai 30 derajat celcius), dan sinyal untuk setiap level kecepatan kipas pada AC (*low, mid, high*, dan *auto*). Data-data sinyal ini nantinya akan digunakan untuk mengontrol AC menggunakan *microcontroller*. Data sinyal berupa lebar waktu untuk setiap bit perintah.

B. Perancangan Arsitektur dan Protokol Sistem

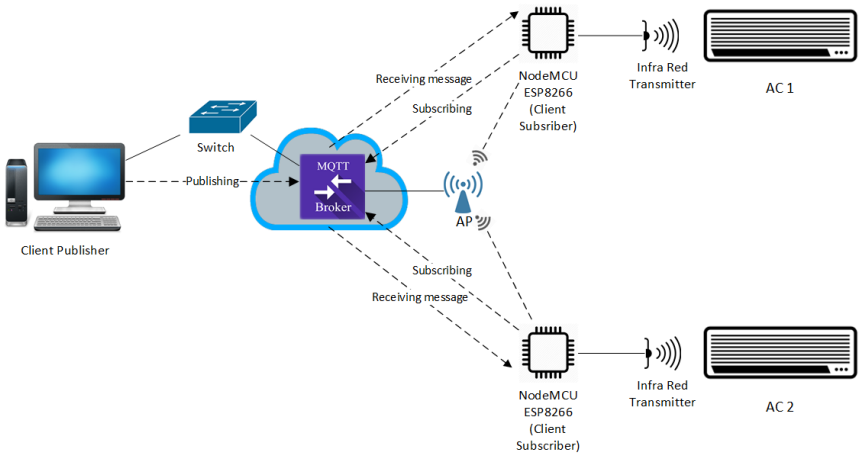
Sistem akan dibangun dengan menerapkan protokol MQTT dimana setiap perangkat yang terhubung dapat memiliki peran sebagai *publisher* dan *subscriber*. Protokol MQTT (*Message Queuing Telemetry Transport*) adalah protokol yang berjalan di atas *stack* TCP/IP dan mempunyai ukuran paket data dengan *overhead* yang kecil *(*minimum 2 bytes) sehingga berdampak pada konsumsi catu daya yang cukup kecil. Menurut [9], protokol MQTT adalah jenis protokol *data-agnostic* yang artinya kita bisa mengirimkan data apapun seperti data *binary*, *text*, bahkan XML ataupun JSON. Protokol ini berbasis *publish/ subscribe* dan dirancang khusus untuk komunikasi dengan karakteristik sederhana, ringan antar perangkat berdaya rendah [10].

1) *Publish dan Subscribe: Publish* adalah proses mengirimkan suatu message/data ke broker pada suatu topic tertentu. Sedangkan *subscribe* adalah suatu langkah yang dilakukan agar dapat menerima message yang dikirimkan oleh publisher pada topic yang telah ditentukan. Beberapa client dapat melakukan subscribe ke topik yang sama dan mengolah informasi yang didapat sesuai alur program. MQTT memiliki 2 jenis peranan, yaitu *publisher* dan *subscriber*. *Publisher* berperan mengirimkan pesan ke broker pada suatu topik tertentu, yang nantinya akan diterima oleh para *subscriber* yang melakukan *subscribe* pada topik yang sesuai. *Subscriber* berperan menerima pesan yang dikirimkan oleh *publisher* dan memprosesnya.
2) *Broker:* Semua client, baik publisher maupun subscriber harus terkoneksi ke broker untuk dapat mengirim dan menerima data. Menurut [11], broker bertugas menerima pesan/data yang dikirimkan oleh *publisher/* pengirim, menyaring pesan-pesan yang masuk, dan meneruskannya ke *subscriber* sebagai penerima pesan pada topic yang sesuai. Broker juga berperan sebagai *space decoupling* dan yang lebih penting lagi yaitu *time decoupling*, dimana *publisher* dan *subscriber* tidak perlu terkoneksi secara bersamaan, misalnya subscriber bisa saja disconnect setelah melakukan subscribe ke broker dan beberapa saat kemudian *subscriber* terhubung kembali ke broker dan *subscriber* tersebut tetap akan menerima data yang tertunda sebelumnya.
3) *Topic/Subscription:* Pada protokol MQTT, suatu pesan pasti dikirimkan oleh *publisher* dalam suatu topik tertentu. Topik memiliki sistem hierarki yang dipisahkan dengan tanda " / " (slash). Hal ini memungkinkan adanya pengelompokan seperti penyimpanan file dalam komputer yang dikelompokkan dalam berbagai folder. *Subscriber* harus membuat suatu *subscription* ke suatu topik tertentu untuk dapat menerima pesan yang dikirimkan oleh *publisher* pada topik tersebut. Contoh penulisan topik: house/living-room/main-light, house/room1/AC, house/1st-floor/living-room/TV, dan sebagainya.

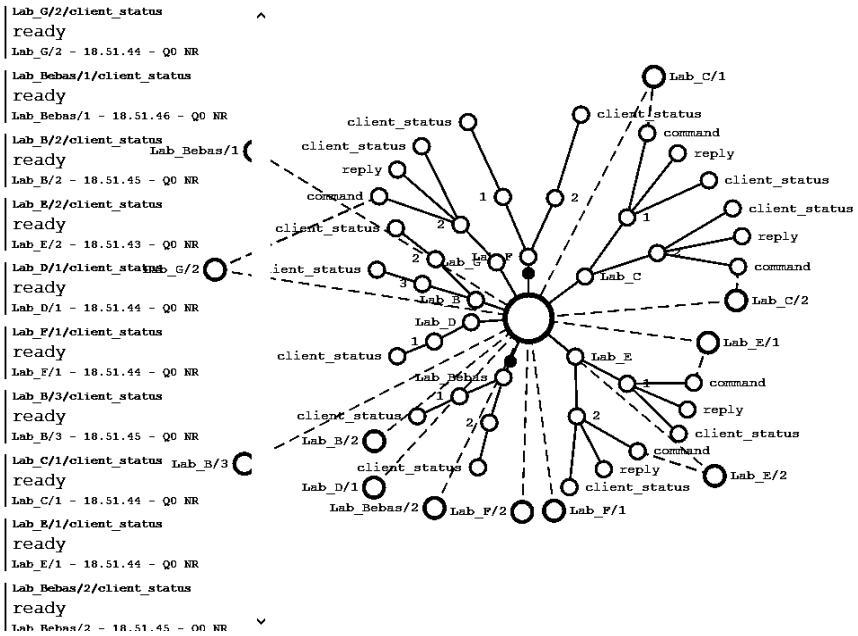
Sistem yang akan dibangun akan dikonfigurasi dengan 1 *client publisher* dan banyak *client subscriber*. *Client publisher* berupa 1 perangkat komputer yang akan dipasang suatu aplikasi yang dibuat dalam penelitian ini, sedangkan *client subscriber* berupa *microcontroller* NodeMCU ESP8266 yang terhubung ke transmitter inframerah. *Client publisher* ditempatkan di ruang petugas laboratorium, sedangkan *client subscriber* dipasang pada setiap AC di seluruh ruang laboratorium. Baik *client publisher* maupun *client subscriber* terhubung ke *broker* melalui Internet. *Broker* yang digunakan adalah shiftr.io. *Client publisher* terhubung ke Internet melalui kabel LAN. Sedangkan *client subscriber* terhubung ke Internet melalui jaringan Wi-Fi yang ada di laboratorium. Topologi sistem yang dibuat dapat dilihat pada Gambar 2.

Setiap perangkat AC akan memiliki topiknya masing-masing dalam berkomunikasi dengan *broker* sesuai struktur topik yang dirancang pada protokol MQTT. Topik untuk setiap AC terdiri dari topik *command, topic reply,* dan topik *client status.* Ketiga topik tersebut disimpan pada *database* pada saat melakukan tambah data AC. Secara default, topik *command*

memiliki format: "[nama_ruang]/[nomor_AC]/command", topik *reply* memiliki format: "[nama_ruang]/[nomor_AC]/reply", topik *client status* memiliki format "[nama_ruang]/[nomor_AC]/client_status". Jika nama ruang mengandung spasi, maka akan diganti dengan *underscore*. Misal, AC nomor 1 yang berada di ruang laboratorium A secara *default* memiliki topik *command* Lab_A/1/command, topik *reply* Lab_A/1/reply, dan topik *client status* Lab_A/1/client_status. Topik tersebut dapat diganti sesuai kebutuhan. Rancangan struktur topik yang akan digunakan diperlihatkan pada Gambar 3.



Gambar 2. Topologi Sistem



Gambar 3. Struktur topik protocol MQTT

C. Perancangan Aplikasi Desktop Penjadwalan

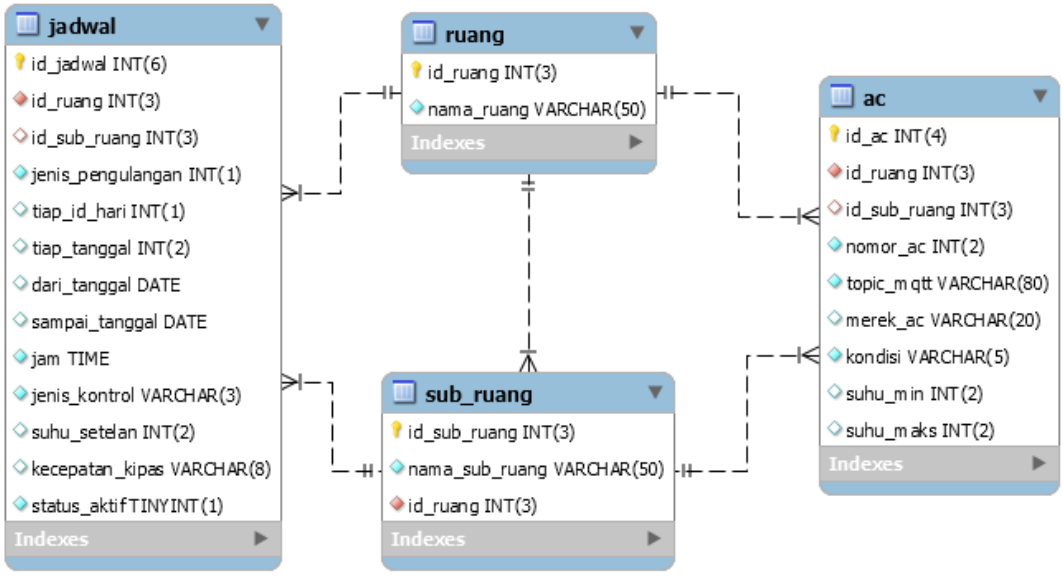
Aplikasi *desktop* dibangun dengan bahasa pemrograman Visual Basic .NET dan *database server* MySQL. Aplikasi tersebut nantinya akan berperan sebagai *client publisher*. Aplikasi dibuat dengan adanya beberapa panel/ tombol untuk menghidupkan, mematikan, mengubah temperature, dan mengubah kecepatan kipas pada AC secara manual, sehingga pengguna dapat melakukan kontrol manual pada AC di seluruh ruang laboratorium komputer Agape lantai 2 Universitas

Kristen Duta Wacana. Jika aplikasi tersebut menerima suatu perintah dari pengguna, misalnya perintah untuk menghidupkan AC di ruang laboratorium B, maka aplikasi tersebut akan mengirimkan (*publish*) *message* (perintah untuk menghidupkan AC) ke *broker* pada *topic* yang sesuai untuk AC di ruang tersebut. Format yang sama juga digunakan untuk perintah menghidupkan, mengubah setelan suhu, dan mengubah kecepatan kipas pada AC yaitu: "on/[suhu]/[kecepatan_kipas]", misalnya: "on/20/sedang". Suhu berupa angka dari 16 sampai 30, sedangkan kecepatan kipas memiliki 4 level, yaitu rendah, sedang, tinggi, dan otomatis. Misal *state* AC saat ini yang ada pada panel kontrol AC adalah "Status: OFF, Suhu: 20, Kecepatan Kipas: Tinggi", kemudian user menghidupkan AC dengan menekan tombol ON/OFF sebanyak 1 kali, maka pesan yang dikirimkan adalah "on/20/tinggi" dan *state* pada panel kontrol AC akan berubah status nya menjadi: "Status: ON, Suhu: 20, Kecepatan Kipas: Tinggi". Kemudian user menaikkan setelan suhu dengan mengklik tombol Temp Up sebanyak 1 kali, maka pesan yang dikirimkan adalah "on/21/tinggi". *State* pada panel kontrol AC akan berubah suhu nya menjadi: "Status: ON, Suhu: 21, Kecepatan Kipas: Tinggi". Setelah itu, user mengubah kecepatan kipas menjadi rendah, maka pesan yang dikirimkan adalah "on/21/rendah". Sedangkan ketika mematikan AC, pesan yang dikirimkan adalah "off".

Aplikasi memiliki sebuah *form* untuk memasukkan jadwal untuk menghidupkan atau mematikan AC di suatu ruang laboratorium. Sehingga, aplikasi tersebut juga berperan untuk melakukan penjadwalan AC di seluruh ruang laboratorium berdasarkan jadwal yang terdapat pada *database*. Misalnya jadwal ruang laboratorium D selesai digunakan pada pukul 10.20, maka pada jam tersebut aplikasi bertugas sebagai *client publisher* akan mengirimkan pesan (perintah untuk mematikan AC) ke *broker* pada topik yang sesuai untuk AC di ruang tersebut. Perancangan *database* aplikasi terlihat pada Gambar 4Gambar 4.

Database memiliki 4 tabel, yaitu tabel ruang, sub ruang, AC, dan jadwal. Satu ruang dapat memiliki banyak sub ruang, sedangkan satu sub ruang terdapat pada satu ruang tertentu. Sehingga tabel ruang memiliki relasi *one to many* terhadap tabel *sub ruang*. Satu ruang / sub ruang dapat memiliki lebih dari 1 AC, sedangkan 1 AC terdapat pada satu ruang / sub ruang tertentu. Sehingga tabel ruang dan sub ruang memiliki relasi *one to many* terhadap tabel AC. Satu ruang atau sub ruang bisa tercantum pada banyak jadwal, sedangkan satu jadwal hanya untuk satu ruang atau sub ruang tertentu. Sehingga tabel ruang dan sub ruang memiliki relasi *one to many* terhadap tabel jadwal. Tabel jadwal tidak berelasi langsung terhadap tabel AC, namun berelasi terhadap tabel ruang dan sub ruang. Sehingga penjadwalan berhubungan dengan pemakaian ruang dan sub ruang. Dengan begitu, seluruh AC yang terdapat pada ruang / sub ruang yang tercantum pada jadwal tertentu akan terlibat dalam penjadwalan hidup / mati nya AC. Misal, ada jadwal untuk ruang laboratorium A pada hari Senin jam 17.30 dengan jenis kontrol hidupkan, maka seluruh AC di ruang laboratorium A pada hari Senin jam 17.30 akan menyala secara otomatis.

Topik MQTT untuk setiap AC di semua ruang tersimpan pada *database*. Ketika pengguna mengirimkan perintah melalui Panel Kontrol AC atau ada jadwal yang bersesuaian dengan jam saat itu, maka aplikasi pada *client publisher* akan mengirimkan / mem*publish* pesan ke broker pada topik *command* sesuai AC yang dituju.



Gambar 4. Rancangan Basis Data Aplikasi

D. Perancangan Perangkat Modul Inframerah

Perangkat modul inframerah yang akan dipasang pada AC dibangun menggunakan program pada Arduino IDE. Modul inframerah ini menggunakan *microcontroller* NodeMCU ESP8266 yang akan berperan sebagai *client subscriber*. Program yang dibuat untuk *client subscriber* ini berfungsi untuk melakukan *subscribe* ke topik *command* untuk setiap AC, menerima *message* yang dikirimkan oleh *client publisher*, dan memerintahkan *transmitter* inframerah untuk mengeluarkan sinyal inframerah sesuai perintah pada *message* yang diterimanya. Pada setiap AC akan dipasang 1 modul inframerah untuk mengontrol AC tersebut.

Modul inframerah yang terpasang pada AC terdiri *microcontroller* NodeMCU ESP8266 version 3, LED inframerah, LED warna merah, dan LED warna hijau yang tersusun pada papan *circuit board*. NodeMCU ESP8266 adalah sebuah *microcontroller* yang dibuat oleh Espressif dan menjadi salah satu primadona baru di dunia IoT karena harganya yang relatif sangat murah. NodeMCU ESP8266 berukuran panjang 4,83 cm, lebar 2,54 cm, dan berat sekitar 7 gram. Modul NodeMCU ESP8266 tersebut terdapat GPIO, PWM, I2C, dan ADC semua dalam satu board [12]. Board ini juga sudah dilengkapi dengan fasilitas Wi-Fi dan firmware nya bersifat open source. Gambar 3 adalah pin mapping ESP8266.

LED inframerah akan diarahkan ke sensor inframerah yang terdapat pada AC. LED warna merah akan menyala ketika *microcontroller* belum terhubung ke *broker* atau koneksi ke *broker* terputus. Sedangkan LED warna hijau akan menyala ketika *microcontroller* berhasil terhubung ke *broker*. LED inframerah terhubung ke pin D2, LED warna merah terhubung ke pin D3, sedangkan LED warna hijau terhubung ke pin D4.

Modul inframerah yang akan menjadi c*lient subscriber* akan terpasang pada AC dan diprogram untuk melakukan *subscribe* ke topik *command* sesuai topik *command* yang ada pada aplikasi *client publisher*. Sehingga, ketika *client publisher* mengirimkan pesan ke suatu topik *command* untuk AC tertentu, maka *client subscriber* pada AC yang dituju akan menerima pesan yang dikirimkan oleh *client publisher*. Sebagai contoh, pada aplikasi *client publisher* AC nomor 1 yang berada di ruang laboratorium B diberi topik *command default*, yaitu "Lab_B/1/command". *Client subscriber* yang terpasang pada AC nomor 1 di ruang laboratorium B diprogram untuk melakukan *subscribe* ke topik *command* "Lab_B/1/command" juga. Pada saat mematikan AC tersebut, *client publisher* akan mengirimkan pesan "off" ke topik "Lab_B/1/command". Sehingga, *client subscriber* pada AC tersebut akan menerima pesan yang dikirimkan oleh *client publisher*, dan AC pun akan mati.

Setelah *client subscriber* menerima perintah yang dikirimkan oleh *client publisher* dan mengirim sinyal ke AC yang dituju, *client subscriber* diprogram untuk mengirimkan *reply* / balasan berupa pesan "command received" ke topik *reply* untuk AC tersebut agar *client publisher* dapat mengetahui apakah perintah berhasil diterima oleh *client subscriber* yang dituju atau tidak. Ketika pengguna membuka Panel Kontrol AC, aplikasi *client publisher* akan melakukan *subscribe* ke topik *reply*. Sehingga ketika pengguna mengirimkan perintah ke suatu AC, dan aplikasi *client publisher* menerima pesan "command received" dari *client subscriber* pada AC yang dituju, maka akan muncul tulisan "Perintah Terkirim" di bagian bawah Panel Kontrol AC.

Jika *client subscriber* sedang *online* / terhubung ke broker, maka *client subscriber* akan mengirim/mem*publish* pesan "ready" ke topik *client_status* setiap 3 detik sekali. Sedangkan *client publisher* akan melakukan *subscribe* ke topik *client_status* untuk memantau status *client subscriber* pada setiap AC secara *realtime* (*client subscriber* sedang *online* atau *offline*). Jika dalam waktu 9 detik *client publisher* tidak menerima pesan "ready" dari suatu *client subscriber*, maka *client subscriber* tersebut dianggap sedang *offline* atau koneksinya ke broker sedang terputus.

E. Perancangan Pengujian dan Evaluasi Sistem

Pada tahap ini, sistem akan diuji dan dievaluasi untuk pengembangan selanjutnya. Aspek yang akan dievaluasi antara lain:

1) *Tingkat keberhasilan kontrol manual:* Akan dilakukan pengujian tingkat keberhasilan sistem menjalankan perintah yang diberikan oleh pengguna melalui Panel Kontrol AC. Pengujian ini akan dilakukan terhadap 10 AC yang terpasang IR modul dengan jumlah pengujian sebanyak 8 kali untuk setiap AC (menghidupkan AC sebanyak 2 kali, mematikan AC sebanyak 2 kali, menaikkan suhu sebanyak 1 kali, menurunkan suhu sebanyak 1 kali, mengubah kecepatan kipas sebanyak 2 kali).
2) *Tingkat keberhasilan penjadwalan:* Akan dilakukan pengujian terhadap tingkat keberhasilan sistem penjadwalan yang akan dibuat. Pengujian ini akan dilakukan terhadap 30 jadwal dengan status aktif pada hari, jam, ruang, dan jenis kontrol yang berbeda-beda.
3) *Waktu delay kontrol manual:* Akan dilakukan pengukuran waktu delay yang terjadi antara pengguna memberikan perintah melalui Panel Kontrol AC dan AC yang dituju menanggapi perintah tersebut. Pengujian ini akan dilakukan terhadap 10 AC yang terpasang IR modul dengan jumlah pengujian sebanyak 8 kali untuk setiap AC.
4) *Waktu delay penjadwalan:* Akan dilakukan pengukuran waktu delay terhadap sistem penjadwalan yang akan dibuat. Pengujian ini akan dilakukan terhadap 30 jadwal dengan status aktif pada hari, jam, ruang, dan jenis kontrol yang berbeda-beda.

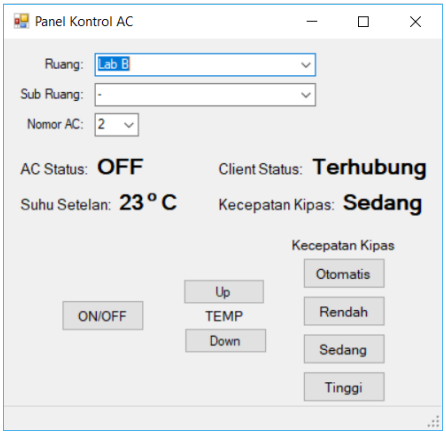
p-ISSN : 2443-2210
e-ISSN : 2443-2229

III. HASIL DAN PEMBAHASAN

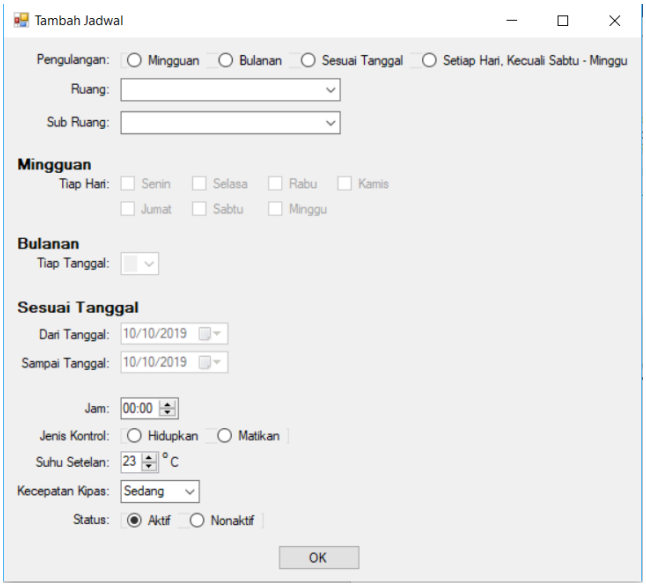
A. Hasil Implementasi Aplikasi untuk Client Publisher

Hasil tampilan Panel Kontrol AC aplikasi untuk *client publisher*, seperti pada Gambar 5. Untuk dapat mengontrol AC, pengguna harus memilih ruang dan nomor AC yang akan dikontrol terlebih dahulu. Melalui panel kontrol AC, pengguna dapat menghidupkan dan mematikan AC, serta mengubah suhu setelan dan kecepatan kipas pada AC. Setelan suhu hanya dapat dinaikkan dan diturunkan hingga batas yang terdapat pada data AC yang bersangkutan. Kecepatan kipas terdiri dari 4 level, yaitu otomatis, rendah, sedang, dan tinggi. Jika status AC dalam keadaan OFF, apabila pengguna mengubah suhu setelan maupun kecepatan kipas, sistem tidak akan mengirim perintah apapun ke AC yang dituju. *State* yang terdapat pada panel kontrol AC hanya menunjukkan perintah yang dikirim ke AC ketika pengguna mengklik tombol yang ada di panel kontrol AC tersebut, bukan *state* dari AC yang bersangkutan secara *realtime*. Karena apabila AC dikontrol menggunakan remote konvensional, *state* yang terdapat pada panel kontrol AC tidak akan berubah.

Gambar 6 adalah tampilan *form* Tambah Jadwal. Ketika melakukan tambah jadwal, pengguna perlu memilih jenis pengulangan yang diinginkan, memilih ruang dan sub ruang, memilih hari atau tanggal, memasukkan jam, memilih jenis kontrol (hidupkan atau matikan), dan status jadwal (aktif atau nonaktif). Jika pengguna memilih jenis kontrol hidupkan, maka pengguna perlu mengisi suhu setelan dan memilih kecepatan kipas yang diinginkan. Jika status jadwal aktif, maka pada hari/tanggal yang dipilih dan pada jam yang diisikan, sistem akan mengirim perintah ke AC yang terdapat pada ruang / sub ruang yang terpilih. Akan tetapi, jika status jadwal nonaktif, maka sistem akan mengabaikan jadwal tersebut.



Gambar 5. Panel Kontrol AC



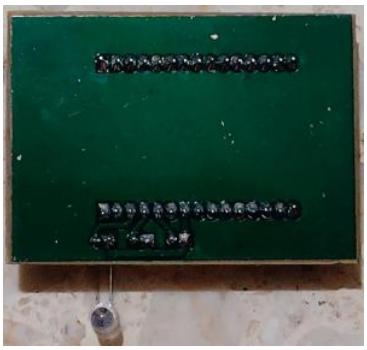
Gambar 6. *Form* Tambah Jadwal

B. Hasil Perakitan IR Modul

IR modul terdiri dari *microcontroller* NodeMCU ESP8266, LED inframerah, LED warna merah, dan LED warna hijau yang disolder pada PCB. PCB tersebut berukuran 7 cm x 4,5 cm. Pada Gambar 7, urutan LED yang terpasang dari sebelah kiri adalah LED warna hijau, LED warna merah, dan LED inframerah. LED warna merah akan menyala ketika modul tersebut pertama kali dinyalakan atau koneksi ke *broker* terputus. Sedangkan LED warna hijau akan menyala ketika *microcontroller* telah berhasil terhubung ke *broker* melalui jaringan WiFi. LED inframerah berfungsi untuk mengirim perintah ke AC. Kaki LED inframerah ditekuk seperti pada gambar agar saat dipasang pada AC dapat mengarah ke sensor inframerah yang terdapat pada AC

Hasil perakitan IR Modul dapat dilihat pada Gambar 7 dan Gambar 8.

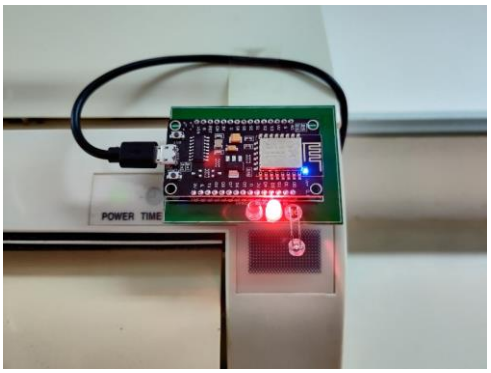
Gambar 7. IR Modul Tampak Atas



Gambar 8. IR Modul Tampak Bawah

C. Hasil Pemasangan IR Modul pada AC

Hasil pemasangan IR modul pada AC dapat dilihat pada Gambar 9 dan Gambar 10.



Gambar 9. IR Modul Terpasang pada AC dengan Kondisi Belum Terhubung ke Broker

Gambar 10. IR Modul Terpasang pada AC dengan Kondisi Telah Terhubung ke Broker

Gambar 9 adalah hasil pemasangan IR modul pada AC laboratorium F. Modul tersebut ditempel pada AC dengan LED inframerah mengarah tepat ke sensor inframerah yang terdapat pada AC tersebut. Modul tersebut diberi daya listrik melalui kabel micro USB yang tersambung ke adaptor 5 Volt. Pada Gambar 9, LED yang menyala adalah LED warna merah. Hal tersebut menandakan bahwa *microcontroller* belum terhubung ke *broker* atau koneksi ke *broker* sedang terputus. adalah hasil pemasangan IR modul pada AC laboratorium bebas. Pada Gambar 10, LED yang menyala adalah LED warna hijau. Hal ini menandakan bahwa *microcontroller* telah terhubung ke *broker.*

TABEL 1
HASIL PENGUJIAN SISTEM KONTROL MANUAL PADA AC

Ruang / Perintah	Waktu *delay* terhadap respon perintah dalam detik (s)									
	Ruang Laboratorium									
	B		C	D	Bebas		E	F	G	
	AC 2	AC 3	AC 1	AC 1	AC 1	AC 2	AC 2	AC 1	AC 2	AC 1
Hidupkan	2.84	3.44	2.48	2.84	2.45	3.15	2.38	3.78	1.69	2.20
Hidupkan	3.42	2.45	-	2.87	3.91	4.97	1.94	4.03	1.96	4.20
Matikan	4.28	2.38	3.09	2.56	1.67	3.60	2.71	4.49	4.08	2.88
Matikan	3.07	2.01	2.20	3.75	2.96	3.84	4.16	1.93	1.80	3.34
Naikkan Suhu	3.21	2.51	3.48	1.88	3.08	2.23	2.99	1.80	2.98	4.91
Turunkan Suhu	3.39	2.88	2.82	4.27	3.65	4.84	3.42	3.16	3.95	2.11
Ubah Kecepatan Kipas	2.97	1.87	3.51	2.42	2.47	3.95	5.01	3.82	3.24	2.38
Ubah Kecepatan Kipas	3.54	2.21	3.53	3.85	3.43	2.72	4.35	-	3.46	1.98
Tingkat keberhasilan	97.5 %									
Delay rata-rata	3.10									
Delay minimum	1.67									
Delay maksimum	5.01									

D. Hasil Pengujian Sistem Kontrol Manual

Hasil pengujian sistem kontrol manual pada AC dapat dilihat pada Tabel 1. Dari hasil pengujian yang tertera pada Tabel 1 dapat disimpulkan bahwa sistem kontrol manual pada AC yang dibuat pada penelitian ini dapat bekerja dengan baik. Tingkat keberhasilan yang didapatkan yaitu sebesar 97,5% dan waktu *delay* rata-rata yang relatif singkat, yaitu sebesar 3,1 detik. Pada hasil yang didapat di Tabel 1, ada dua proses pengiriman perintah control yang tidak direspon oleh perangkat AC. Hal ini disebabkan karena perangkat IR Modul seketika tidak terhubung dengan broker, sehingga proses pengiriman pesan menjadi gagal. Berdasarkan hasil tersebut, dapat disimpulkan bahwa protokol MQTT adalah protokol yang handal untuk mengirim pesan ke *microcontroller* dan *broker* yang digunakan, yaitu shiftr.io merupakan salah satu *broker online* yang cukup baik, karena memiliki tingkat keberhasilan pengiriman pesan yang tinggi.

E. Hasil Pengujian Sistem Penjadwalan AC

Hasil pengujian sistem penjadwalan AC dapat dilihat pada Tabel 2.

TABEL 2
HASIL PENGUJIAN SISTEM PENJADWALAN AC

No.	Ruang	Hari	Jam	Jenis Kontrol	Berhasil (Ya/Tidak)	Waktu *delay* (s)
1	Lab B	Senin	08.10	Hidupkan	Ya	12,7
2	Lab G	Senin	11.15	Hidupkan	Ya	14,3
3	Lab E	Senin	14.15	Hidupkan	Ya	15,0
4	Lab D	Senin	14.20	Matikan	Ya	11,5
5	Lab G	Senin	17.15	Hidupkan	Ya	13,2
6	Lab Bebas	Senin	20.00	Matikan	Ya	15,6
7	Lab C	Selasa	07.15	Hidupkan	Tidak	-
8	Lab F	Selasa	07.15	Hidupkan	Ya	11,4
9	Lab D	Selasa	13.20	Matikan	Ya	8,9
10	Lab B	Selasa	19.20	Matikan	Ya	10,1
11	Lab E	Selasa	19.20	Matikan	Ya	12,3
12	Lab G	Rabu	07.15	Hidupkan	Ya	16,8
13	Lab Bebas	Rabu	07.15	Hidupkan	Tidak	-
14	Lab F	Rabu	10.15	Hidupkan	Ya	17,1
15	Lab D	Rabu	10.20	Matikan	Ya	15,7
16	Lab B	Rabu	13.15	Hidupkan	Ya	14,2
17	Lab F	Rabu	13.20	Matikan	Ya	16,4
18	Lab G	Rabu	16.20	Matikan	Ya	15,9
19	Lab C	Rabu	19.20	Matikan	Ya	14,0

No.	Ruang	Hari	Jam	Jenis Kontrol	Berhasil (Ya/Tidak)	Waktu *delay* (s)
20	Lab D	Kamis	07.15	Hidupkan	Ya	9,2
21	Lab E	Kamis	07.15	Hidupkan	Ya	7,5
22	Lab B	Kamis	13.20	Matikan	Ya	7,9
23	Lab C	Kamis	16.15	Hidupkan	Ya	8,3
24	Lab E	Kamis	16.20	Matikan	Ya	9,8
25	Lab D	Kamis	19.20	Matikan	Ya	8,1
26	Lab G	Jumat	07.15	Hidupkan	Ya	12,6
27	Lab E	Jumat	10.15	Hidupkan	Tidak	-
28	Lab F	Jumat	10.20	Matikan	Ya	10,7
29	Lab B	Jumat	13.15	Hidupkan	Ya	13,2
30	Lab E	Jumat	16.20	Matikan	Ya	11,4

Tingkat keberhasilan	90%
Waktu *delay* rata-rata	12,36
Waktu *delay* minimum	7,5
Waktu *delay* maksimum	17,1

Dari hasil pengujian yang tertera pada Tabel 2 dapat dilihat bahwa tingkat keberhasilan penjadwalan AC yang dibuat dalam penelitian ini cukup tinggi, yaitu sebesar 90% dan waktu *delay* rata-ratanya tergolong relatif singkat, yaitu sebesar 12,36 detik. Kegagalan yang terjadi disebabkan ketika ada jadwal yang bersesuaian dengan jam saat itu, *microcontroller* yang terpasang pada AC sedang tidak terhubung ke broker atau koneksinya sedang terputus. Tetapi, tingkat kegagalan yang terjadi relatif kecil, yaitu hanya sebesar 10% saja dan waktu *delay* rata-rata nya cukup singkat. Sehingga, dapat disimpulkan bahwa sistem penjadwalan AC yang dibuat dalam penelitian ini dapat bekerja dengan baik.

IV. SIMPULAN

Dari hasil penelitian yang dilakukan, dapat disimpulkan bahwa sistem kontrol dan penjadwalan AC yang dibuat dalam penelitian ini berhasil diimplementasikan di laboratorium komputer Agape lantai 2 UKDW dan dapat bekerja dengan baik dengan performa yang cukup memuaskan. Sistem kontrol manual yang dibuat memiliki tingkat keberhasilan yang cukup tinggi, yaitu sebesar 97,5% dengan waktu *delay* rata-rata yang relatif singkat, yaitu 3,1 detik. Sedangkan sistem penjadwalan yang dibuat memiliki tingkat keberhasilan yang cukup tinggi juga, yaitu sebesar 90% dan waktu *delay* rata-rata yang masih tergolong relatif singkat, yaitu 12,36 detik (tidak sampai setengah menit). Dengan demikian, dapat disimpulkan bahwa protokol MQTT adalah protokol yang handal untuk mengirim pesan dari PC ke *microcontroller* atau sebaliknya, terbukti sistem yang dibuat dapat bekerja dengan baik.

Dengan diimplementasikannya sistem penjadwalan AC tersebut, petugas laboratorium komputer tidak perlu lagi menghidupkan dan mematikan AC secara manual satu per satu setiap harinya, karena sistem yang akan melakukannya secara otomatis sesuai jadwal yang telah diinputkan. Dengan demikian dapat menghemat waktu dan energi. Selain itu, dengan diimplementasikannya sistem penjadwalan AC tersebut dapat meminimalkan pemborosan listrik, karena AC akan mati secara otomatis ketika ruang laboratorium sudah tidak digunakan.

Dari hasil implementasi, sistem yang telah dibuat masih memiliki beberapa kekurangan, yaitu pengguna tidak dapat mengetahui status/kondisi AC secara *realtime*, apakah AC sedang dalam kondisi menyala atau mati, karena tidak ada sensor yang digunakan untuk mendeteksinya. Dalam hal keamanan sistem, tidak ada mekanisme keamanan yang diimplementasikan pada sistem tersebut, sehingga jika ada yang mengetahui username dan password MQTT untuk sistem ini beserta topik untuk setiap AC nya, maka AC dapat dikontrol dengan mengirimkan pesan secara manual ke topik AC yang dituju. Dari sisi pencatatan aktivitas pengendalian perangkat AC masih belum ada fitur *logging* untuk sistem penjadwalan yang telah dibuat, sehingga pengguna tidak dapat mengetahui apakah penjadwalan berhasil dijalankan atau tidak. Implementasi source code masih berbeda untuk setiap merek AC yang digunakan, sehingga ketika hendak meng*upload* program ke *microcontroller* harus memerhatikan merek AC nya terlebih dahulu.

Beberapa hal yang masih dapat dikembangkan pada implementasi sistem ini ialah penambahan sensor suhu pada modul inframerah, sehingga sistem dapat mengetahui apakah perangkat AC dalam kondisi menyala atau tidak.

DAFTAR PUSTAKA

[1] A. Junaidi, "Internet Of Things, Sejarah, Teknologi Dan Penerapannya : Review," *J. Ilm. Teknol. Inf.*, vol. IV, no. 3, pp. 62–66, 2015.
[2] R. P. Pratama, "Pengendali Lampu Rumah Berbasis Esp8266 Dengan Protokol Mqtt," *TESLA J. Tek. Elektro*, vol. 22, no. 1, p. 56, 2020.
[3] A. F. Hakim, W. Wedhaswara, and A. Z. Mardiansyah, "Sistem Pendukung Keputusan Penerangan Ruangan Berbasis IoT Menggunakan Protokol MQTT dan Fuzzy Tsukamoto," *J. Teknol. Informasi, Komputer, dan Apl. (JTIKA)*, vol. 2, no. 2, pp. 304–313, 2020.
[4] I. Harjanto, "IoT Gateway Menggunakan Protokol MQTT pada Perangkat Kendali Berbasis Modbus-RTU," *J. Ilm. TeknoSains*, vol. VI, no. 1, pp.

12–19, 2020.
[5] L. K. P. Saputra and Y. Lukito, "Implementation of air conditioning control system using REST protocol based on NodeMCU ESP8266," *Proceeding 2017 Int. Conf. Smart Cities, Autom. Intell. Comput. Syst. ICON-SONICS 2017*, vol. 2018-Janua, pp. 126–130, 2017.
[6] E. Erham, Markus, A. Surjanto, and J. Rukmana, "Design of a new PID controller based on Arduino Uno R3 with application to household refrigerator," *MATEC Web Conf.*, vol. 154, no. 11, pp. 174–182, 2018.
[7] N. V. K. Ramesh, S. V. Tejesh Kumar, V. Vamsi, and S. Akarsh, "Wi-Fi controlled universal remote using ESP8266," *ARPN J. Eng. Appl. Sci.*, vol. 12, no. 24, pp. 7233–7238, 2017.
[8] M. N. Adil and S. Dase, "Infrared Remote Creator Untuk Aplikasi Smart Room Berbasis Mikrokontroler," *Pros. Semin. Nas. Tek. Elektro dan Inform.*, pp. 136–141, 2020.
[9] I. N. Aziza, "Smart Farming Untuk Peternakan Ayam," *J. Teknol. Inf. dan Komun.*, vol. 9, no. 1, pp. 36–40, 2019.
[10] S. O. F. Tarigan, H. I. Sitepu, and M. Hutagalung, "Pengukuran Kinerja Sistem Publish/Subscribe Menggunakan Protokol MQTT (Message Queuing Telemetry Transport) (Publish / Subscribe System Performance Measurement Using the MQTT (Message Queuing Telemetry Transport Protocol)," *J. Telemat.*, vol. 9, no. 1, pp. 25–30, 2014.
[11] C. F. Permatasari and H. Dhika, "Optimasi Jalur Transfer Data dari HTTP menjadi MQTT pada IoT menggunakan Cloud Services," *JISA(Jurnal Inform. dan Sains)*, vol. 1, no. 2, pp. 67–72, 2018.
[12] A. Satriadi, Wahyudi, and Y. Christiyono, "Perancangan Home Automation Berbasis NodeMCU," *Transient*, vol. 8, no. 1, pp. 64–71, 2019.