

# Implementasi Platform Otomasi Rumah OpenHAB pada *Google Cloud Platform*

Muhammad Fadhil Arkan  
Program Studi Teknik Fisika  
Institut Teknologi Bandung  
40132 Bandung, Indonesia  
fadhilarkan@students.itb.ac.id

Rezky Mahesa Nanda  
Program Studi Teknik Fisika  
Institut Teknologi Bandung  
40132 Bandung, Indonesia  
rezky.mahesa.nanda@s.itb.ac.id

**Abstrak**— Kemajuan teknologi telah membuat hidup manusia menjadi lebih mudah. Salah satu teknologi yang menjanjikan adalah *Internet of Things* (IoT). IoT memanfaatkan internet secara lebih spesifik untuk bertukar data antara "*Things*", mengendalikan, dan memantaunya. Sistem otomasi rumah adalah contoh penerapan IoT. Penghuni dapat mengontrol dan memantau rumah secara jarak jauh melalui platform otomasi rumah. Namun kompleksitas secara teknis muncul ketika sistem menjadi lebih besar. Platform sistem otomasi rumah yang terpasang secara *on-premise* dapat mengalami kegagalan ketika perangkat keras tempat ia terpasang mengalami kesalahan. Untuk mengatasi hal tersebut, pada penelitian ini, kami mengajukan sebuah metode untuk mengimplementasikan platform otomasi rumah di layanan cloud menggunakan Google Cloud Platform. Hasil penelitian menunjukkan penggunaan OpenHAB sebagai platform otomasi rumah di Google Cloud Platform cocok untuk sistem dengan kompleksitas tinggi, namun akan memakan biaya untuk sistem sederhana. Karenanya, dapat digunakan layanan Google Cloud Platform secara terpisah. Antarmuka pengguna menggunakan HABPanel ditujukan bagi pengguna yang ingin berkomunikasi dua arah dengan perangkat, sedangkan Google Data Studio dapat dijadikan opsi pendukung apabila data analitik juga ditekankan.

**Kata kunci**—IoT, platform sistem otomasi rumah, google cloud platform

## I. PENDAHULUAN

Perkembangan teknologi mempermudah hidup manusia. Banyak hal dapat dilakukan hanya dalam satu sentuhan jari, mulai memesan makanan hingga mengontrol perangkat secara jarak jauh. Kemungkinan-kemungkinan tersebut ada karena internet.

Merupakan pemanfaatan lebih jauh dari internet, *Internet of Things* (IoT) memungkinkan pengguna untuk melakukan pertukaran data, kontrol, pengawasan, dan lain-lain. "*Things*" pada IoT mengacu kepada sensor, sistem tertanam, aktuator, dan sebagainya. *Things* kemudian dapat saling terhubung melalui jaringan internet. Salah satu contoh penerapan teknologi IoT adalah sistem otomasi rumah. Sistem otomasi rumah memberikan keleluasaan bagi penghuni untuk melakukan kontrol dan pengawasan rumahnya dalam jarak jauh. Tujuan utama dari sistem otomasi rumah adalah mencapai kenyamanan, keamanan, dan efisiensi penggunaan energi. Berbagai perangkat cerdas yang ada di dalam rumah dapat

saling berkomunikasi bertukar data dan dikendalikan melalui jaringan internet. Perangkat tersebut dikendalikan melalui sebuah *platform* otomasi rumah yang dipasang pada gawai/komputer/gateway.

Suatu masalah yang dihadapi ketika mengimplementasikan *platform* otomasi rumah secara *on-premise* adalah keandalan teknis ketika sistem otomasi rumah semakin kompleks. Probabilitas terjadinya kegagalan keseluruhan sistem karena kesalahan suatu sub-sistem cukup besar karena tiap sub-sistem saling terkait. [1] Permasalahan tersebut dapat diatasi dengan mengimplementasikan platform pada layanan *cloud*. Ketika *platform* diimplementasikan secara *cloud*, setidaknya probabilitas kegagalan kontrol dan pemantauan keseluruhan sistem akibat perangkat di mana *platform* dipasang gagal dapat dikurangi.

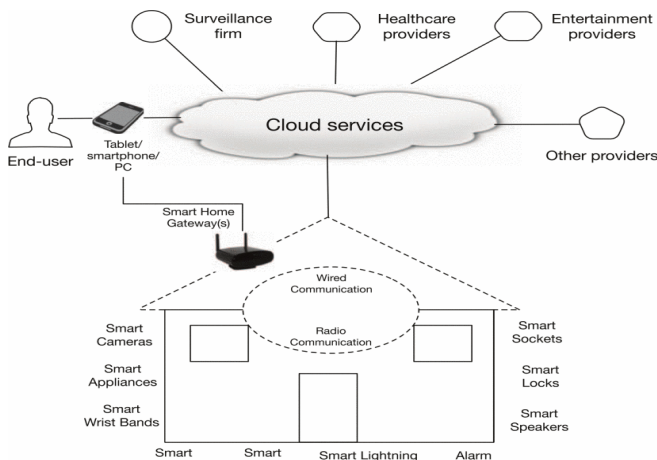
## II. INTERNET OF THINGS (IOT)

*Internet of Things* (IoT) merupakan sebuah konsep yang merepresentasikan berbagai perangkat elektronik ataupun perangkat non-elektronik yang terhubung satu sama lain dan dapat berkomunikasi melalui internet. Berkembangnya IoT difasilitasi oleh berkembangnya teknologi *wireless short-range* yang memungkinkan berbagai macam objek untuk dapat terhubung satu sama lain, seperti Zigbee, RFID, BLE, dan lain-lain [2]. Selain itu, semakin berkembangnya teknologi komputer, perangkat lunak, *cloud*, dan protokol komunikasi juga menyebabkan penerapan IoT semakin lama semakin mungkin untuk dilakukan untuk skala yang semakin besar.

Secara Umum, sebuah sistem IoT memiliki beberapa komponen utama, yaitu (1) *end-device* yang terdiri dari sensor/aktuator, dan mikrokontroler untuk akuisisi data ataupun kontrol, (2) Antarmuka pengguna yang berfungsi sebagai sarana untuk pemantauan dan kontrol, dan (3) *Gateway* yang berfungsi sebagai penghubung antara *end-device* dengan Antarmuka pengguna dan juga dapat digunakan sebagai perangkat untuk mengolah data sebelum di proses lebih lanjut di antarmuka pengguna.

### III. SISTEM OTOMASI RUMAH

Sistem otomasi rumah merupakan sebuah sistem yang terdiri atas sub sistem-sub sistem untuk mengendalikan dan memantau hal-hal yang



**Gambar 1** Arsitektur rumah cerdas. Terdiri atas berbagai perangkat cerdas dan sebuah gateway untuk menyambungkan perangkat-perangkat tersebut dengan jaringan luar. [3]

berpengaruh terhadap aspek kenyamanan manusia. Sub sistem tersebut mencakup keamanan, HVAC, pencahayaan, dan hiburan. [3] Tiap-tiap sub sistem dapat mengakses internet untuk memenuhi permintaan penghuni. [4] Penggunaan sistem otomasi rumah adalah awal dari munculnya terminologi rumah cerdas, di mana perangkat saling terhubung dan dapat dikendalikan secara jarak jauh. [5]

Saat ini, penerapan sistem otomasi rumah cukup menjanjikan. Sebuah survei terhadap 14 rumah tangga pengguna sistem otomasi rumah menunjukkan bahwa kepuasan mereka terhadap sistem otomasi rumah terdiri atas 3 hal, yaitu: kenyamanan, ketenangan, dan kendali yang terpusat. [6] Kenyamanan yang dirasakan responden berasal dari kemudahan akses kontrol ke berbagai perangkat, sedangkan ketenangan berasal dari kemampuan sistem otomasi rumah yang memungkinkan penghuni untuk memeriksa pantauan rekaman CCTV di lingkungan rumah secara jarak jauh, dan kendali terpusat pada survei ini adalah keleluasaan penghuni untuk mengendalikan berbagai perangkat dalam satu dasbor antarmuka sistem otomasi rumah. Penerapan sistem otomasi rumah juga dapat memangkas penggunaan energi sebesar 18,70%. [7] Nilai tersebut logis karena sistem otomasi rumah memungkinkan penghuni untuk memeriksa kembali perangkat yang masih menyala namun tidak digunakan, meskipun saat penghuni sedang berada di luar rumah. Gabungan dari keempat manfaat tersebut membuat emosi penghuni lebih baik sehingga produktivitas meningkat. [8]

Terlepas dari banyaknya manfaat yang diperoleh, salah satu tantangan terbesar pada penerapan sistem

otomasi rumah yang ditemukan pada penelitian Sovacool yakni mengenai keandalan teknis. Semakin kompleks sistem otomasi rumah, semakin saling terhubung pula perangkat-perangkatnya. Keterkaitan dan ketergantungan antar-perangkat tersebut menyebabkan keandalan sistem berkurang jika suatu saat terjadi kesalahan pada satu komponen saja, sehingga memengaruhi kemampuan penghuni untuk melakukan fungsi pengawasan dan pengendalian pada rumahnya sendiri. [1]

Secara garis besar, perangkat sistem otomasi rumah terdiri atas sensor, aktuator, *gateway*, dan perangkat cerdas. Perangkat-perangkat tersebut dapat berkomunikasi dengan protokol komunikasi khusus otomasi rumah, seperti KNX, Zigbee, Z-Wave, dan sebagainya, atau juga dapat melalui Wi-Fi, Bluetooth, dan teknologi komunikasi seluler, dan sebagainya. [5] Pada kasus sistem otomasi rumah yang menggunakan berbagai jenis protokol, sebuah *gateway* memiliki peranan penting untuk menyelaraskan jenis data dan mengirimkannya ke jaringan internet.

Untuk menampilkan data dari perangkat maupun mengontrol perangkat, penghuni dapat melakukannya melalui *platform* otomasi rumah. *Platform* tersebut dapat dipasang pada *gateway*, perangkat pengguna, maupun pada *cloud*. Pada penelitian ini, akan diimplementasikan sebuah *platform* otomasi rumah pada *google cloud platform*.

### IV. OPENHAB

OpenHAB adalah sebuah platform yang dapat berperan sebagai antarmuka pengguna dalam suatu sistem otomasi rumah yang fleksibel. Dengan menggunakan openHAB, dapat dilakukan integrasi antar perangkat IoT dan MQTT broker dengan mudah.

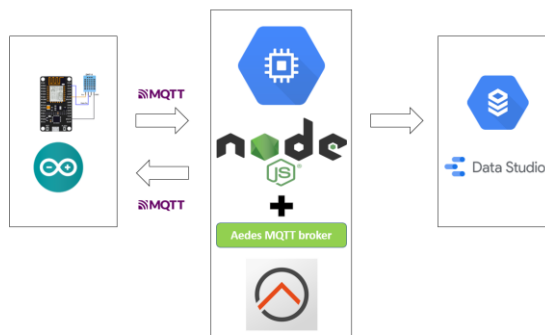
Secara singkat, ada beberapa konsep yang perlu diketahui untuk menggunakan openHAB, di antaranya adalah (1) *Things*, yaitu entitas fisik yang dapat ditambahkan pada sistem seperti saklar lampu, air conditioner, televisi, dan perangkat elektronik lainnya. (2) *Channel*, yaitu fitur yang dimiliki oleh suatu *Thing*, di mana suatu *Thing* bisa memiliki lebih dari satu *thing*. Contohnya adalah sensor DHT yang bisa digunakan mengukur suhu dan kelembapan dapat diakses melalui openHAB melalui *channel* suhu dan *channel* kelembapan, (3) *Items*, yaitu representasi informasi mengenai suatu perangkat, (4) *Bindings*, yaitu *driver* yang berfungsi untuk menghubungkan sistem fisis dengan sistem openHAB, dan (5) *Rules*, yaitu aturan-aturan yang digunakan untuk kebutuhan otomasi.

## V. GOOGLE CLOUD PLATFORM

Dengan semakin berkembangnya teknologi komputasi awan, implementasi untuk suatu sistem IoT pun semakin menjadi sebuah konsep yang menjanjikan. Dengan menggunakan *cloud*, bisa dirancang sebuah server, dan bahkan *gateway* IoT yang bersifat *off-premise*. Oleh karena itu, dengan adanya *cloud* maka proses pemeliharaan perangkat keras untuk server dan *gateway* tidak perlu dilakukan lagi.

Salah satu penyedia platform *cloud* adalah google. Google menyediakan platform *cloud* yang bernama Google Cloud Platform (GCP). Google cloud platform dapat dimanfaatkan sebagai *infrastructure as a service*, *platform as a service*, ataupun sebagai *serverless computing environment*. Beberapa layanan yang disediakan google cloud platform adalah *compute engine*, *storage* dan *database*, platform *big data*, platform *machine learning*, dan masih banyak lagi.

## VI. METODE DAN LANGKAH Pengerjaan



**Gambar 2.** Arsitektur sistem yang digunakan

Pada penelitian ini, digunakan *resource* GCP berupa Google Compute Engine, Google Cloud SQL, dan Google Data Studio seperti yang dideskripsikan pada **Gambar 2**.

Pada Google Compute Engine, dipasang sebuah mesin virtual berbasis linux untuk mengimplementasikan OpenHAB. Pada penelitian ini, OpenHAB diuji coba dengan menerima dan mengirim data dari dan kepada ESP8266, BACnet Simulator, dan AC Daikin yang terpasang di Gedung Labtek VI TP. Rachmat ITB.

Pada laporan ini penulis akan merujuk ESP8266, BACnet Simulator, dan AC Daikin dengan perangkat nomor (1), (2), dan (3) secara berturut-turut.

Komunikasi antara perangkat (1) dan (2) dengan OpenHAB terjadi di atas protokol MQTT, sedangkan

antara perangkat (3) dengan OpenHAB terjadi di atas protokol BACnet.

Perangkat (1) memberikan data berupa temperatur, kelembaban ruangan, status LED, dan status alarm. Data temperatur dan kelembaban diukur di tempat tinggal salah satu penulis yang berlokasi di Kota Bandung, Jawa Barat. Perangkat (2) memberikan data berupa *analog input*, *setting temp (analog output)*, dan *state (multi-state input)*. Perangkat (3) memberikan 28 jenis data, namun pada penelitian ini akan ditampilkan 6 jenis data saja, yakni *set point* temperatur, bacaan sensor temperatur ruangan, status *on/off* AC, mode AC, kecepatan kipas, dan okupansi.

Data-data yang dihimpun oleh OpenHAB dari perangkat-perangkat terdaftar kemudian disimpan ke dalam basis data OpenHAB dan OpenHAB\_SBM di Cloud SQL milik Laboratorium Manajemen Energi, Teknik Fisika ITB dengan tiap jenis data memiliki tabel masing-masing. Visualisasi data dilakukan dengan dua cara, yakni menggunakan HABPanel yang tersedia sebagai salah satu fitur OpenHAB dan Google Data Studio. Setelah divisualisasi, kemudian dianalisis kelebihan dan kekurangan masing-masing *platform* visualisasi data.

### A. Pemasangan OpenHAB di Mesin Virtual

Penulis memasang OpenHAB pada mesin virtual menggunakan langkah pemasangan yang tersedia pada laman web OpenHAB untuk sistem operasi Linux. Langkah pemasangan adalah sebagai berikut:

1. Pada laman VM *instances* pada konsol GCP, lakukan koneksi dengan mesin virtual menggunakan SSH.
2. Kunjungi laman petunjuk pemasangan OpenHAB di [openHAB 2 on Linux | openHAB](#). Karena sistem operasi adalah Linux Ubuntu, maka dipilih sistem pemasangan berbasis apt.
3. Ikuti petunjuk pemasangan untuk OpenHAB versi *Stable Release*.
4. Ketika proses pemasangan selesai, OpenHAB dapat diakses melalui alamat <http://34.121.186.105:8080/>.

### B. Penambahan Perangkat ke OpenHAB

Untuk perangkat (1) dan (2) digunakan MQTT *binding* untuk integrasi dengan OpenHAB. Langkah pemakaian MQTT *binding* adalah sebagai berikut :

1. Buat sebuah MQTT *broker* pada mesin virtual. Pada pekerjaan yang dilakukan, penulis membuat MQTT *broker* menggunakan nodejs dan aedes MQTT *broker*.

2. *Install MQTT binding* melalui menu *add-ons* pada PaperUI
3. Tambahkan *thing* untuk *MQTT broker*, sensor DHT11, dan *BACnet simulator* dan tambahkan *channel* jika ada.
4. Buat *item* untuk masing-masing data yang masuk melalui *things*.
5. Tambahkan *rules* untuk menambahkan fungsi alarm ataupun untuk fungsi otomasi lainnya jika diinginkan.

Untuk perangkat (3) digunakan *BACnet binding* untuk integrasi dengan OpenHAB. Langkah pemakaian *BACnet binding* adalah sebagai berikut :

1. *Install BACnet binding*. *Bacnet binding* belum tersedia pada *add-ons* OpenHAB sehingga harus diunduh terlebih dahulu secara eksternal ke dalam direktori */usr/share/openhab2/addons*. Setelah itu, *BACnet binding* akan tersedia dan dapat di *install* melalui menu *add-ons* pada PaperUI.
2. Tambahkan *thing* untuk *BACnet bridge (Komputer yang menjalankan OpenHAB)*, *BACnet device* (Intesis), dan untuk setiap data yang akan diambil dari *BACnet device (analog input, binary input, multistate input, analog output, binary output, multistate output, analog value, binary value, multistate value)*.
3. Buat *item* untuk masing-masing data yang masuk melalui *things*.
4. Tambahkan *rules* untuk menambahkan fungsi alarm ataupun untuk fungsi otomasi lainnya jika diinginkan.

#### C. Konfigurasi Penyimpanan Data ke Cloud SQL

Penulis menggunakan Cloud SQL sebagai tempat penyimpanan data untuk setiap item yang ada pada OpenHAB. Langkah yang dilakukan untuk integrasi OpenHAB dengan Cloud SQL adalah sebagai berikut :

1. *Install persistence JDBC mysql*. Instalasi *persistence* dilakukan melalui menu *add-ons* pada PaperUI.
2. Konfigurasi *file persistence* pada direktori */etc/openhab2/persistence* untuk menentukan interval *query* data ke Cloud SQL.
3. Konfigurasi *file service* pada direktori */etc/openhab2/service* untuk menentukan lokasi (URL) dari Cloud SQL beserta dengan *username* dan *password* yang digunakan.

4. Konfigurasi zona waktu Cloud SQL melalui menu *flag* pada Cloud SQL GCP.

#### D. Visualisasi Data

Proses menampilkan data pada HABPanel dilakukan dengan langkah berikut:

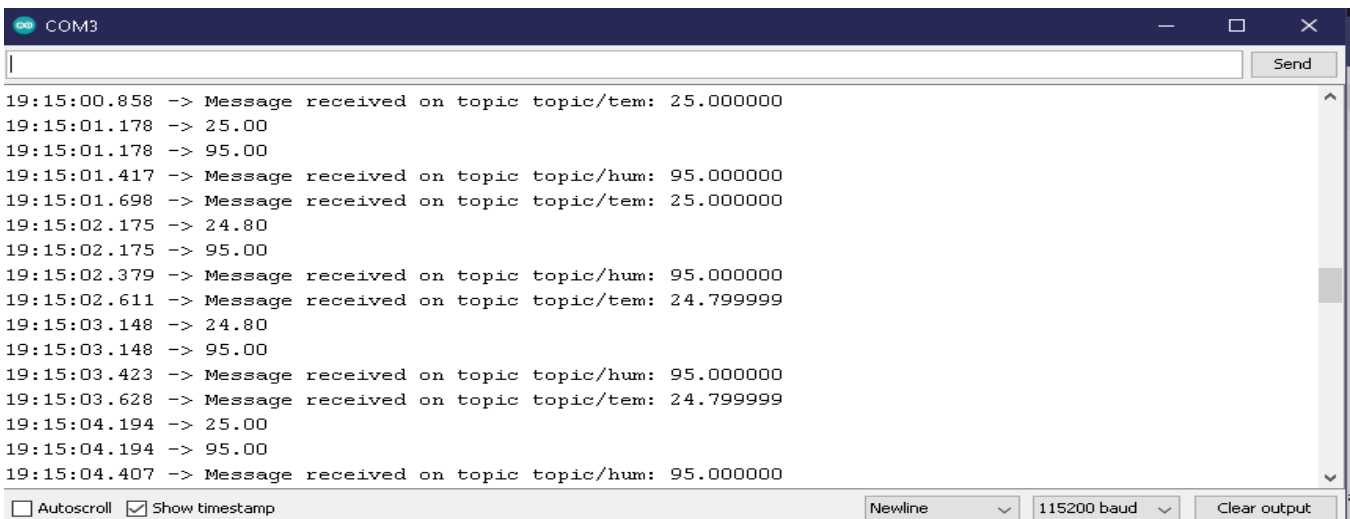
1. Buka HABPanel melalui laman mulai OpenHAB di <http://34.121.186.105:8080/>.
2. Pada sebelah kanan atas, terdapat tombol bersimbol berupa gerigi. Klik tombol tersebut, lalu dipilih “tambahkan dasbor”.
3. Setelah itu, penulis dapat menambahkan *widget* sesuai keinginan.
4. Terdapat 14 jenis *widget* yang tersedia di HABPanel, namun pada semua dasbor perangkat, penulis hanya menggunakan 7 jenis, yaitu: label, jam, tiruan, grafik, *slider*, *switch*, dan knob. Antarmuka perangkat (1), (2), dan (3) terlihat pada **Gambar 7**, **Gambar 11**, dan **Gambar 13** secara berturut-turut.

Visualisasi data di Google Data Studio secara umum dilakukan dengan mengikuti langkah berikut:

1. Pada laman awal Google Data Studio, dipilih opsi membuat laporan kosong.
2. Kemudian, laporan kosong tersebut dihubungkan dengan Cloud SQL di mana data tersimpan. Parameter berikut diperlukan agar Google Data Studio dan Cloud SQL dapat terhubung:
  - Nama koneksi *instance* : tactical-patrol-276605:us-central1:itb-labme
  - *Database* : OpenHAB (Perangkat (1) dan (2)) atau OpenHAB\_SBM (Perangkat (3))
  - Nama pengguna : energy
  - *Password* : energy2x5=10
3. Setelah terhubung, maka dapat ditambahkan diagram untuk menampilkan data. Penulis menggunakan diagram deret waktu, diagram *Gauge*, dan tabel. Penulis juga menyertakan fitur kontrol rentang waktu untuk kustomisasi pengguna. Antarmuka perangkat (1) dan (2) terlihat pada **Gambar 8**. Antarmuka perangkat (3) terlihat pada **Gambar 14**. Pada antarmuka perangkat (1) dan (2), penulis sengaja menambahkan data yang berasal dari luar perangkat yang terhubung dengan OpenHAB, yakni berupa konsumsi energi dan prediksinya.

## VII. HASIL AKHIR

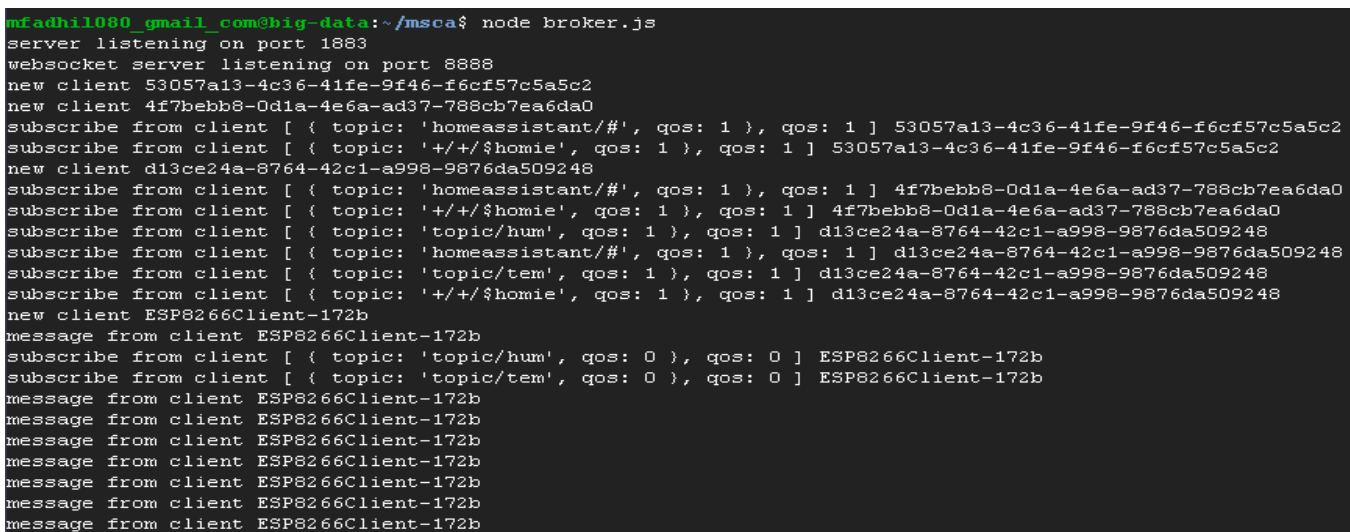
### A. Sensor DHT11+ESP8266



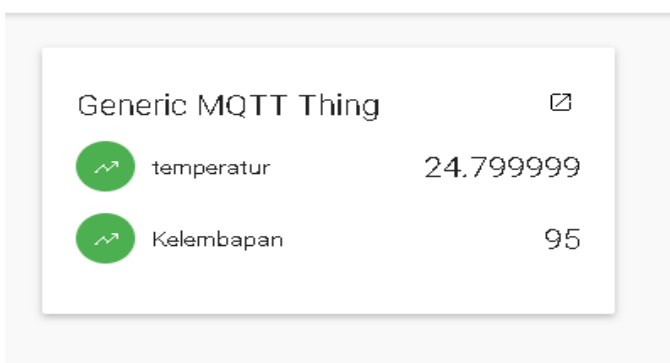
Gambar 3. Serial monitor arduino

Pada **Gambar 3** dapat dilihat bahwa data dari *end-device* (ESP8266 dan sensor DHT11) sudah berhasil terkoneksi dan mengirimkan data ke MQTT broker di mesin virtual GCP.

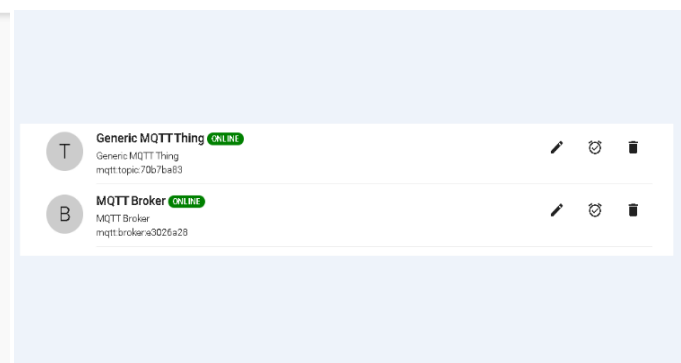
Pada **Gambar 4** dapat dilihat bahwa MQTT broker pada virtual machine di GCP sudah berhasil menerima data dari *end-device*.



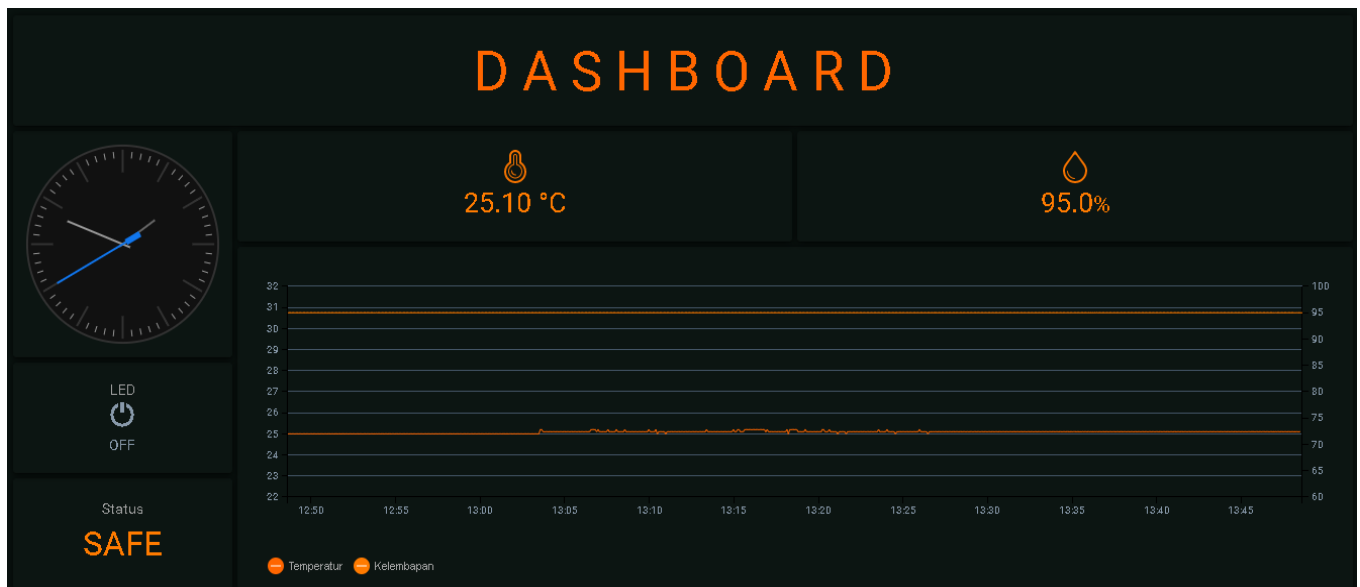
Gambar 4. MQTT broker



Gambar 5. Tampilan data dari sensor di openHAB



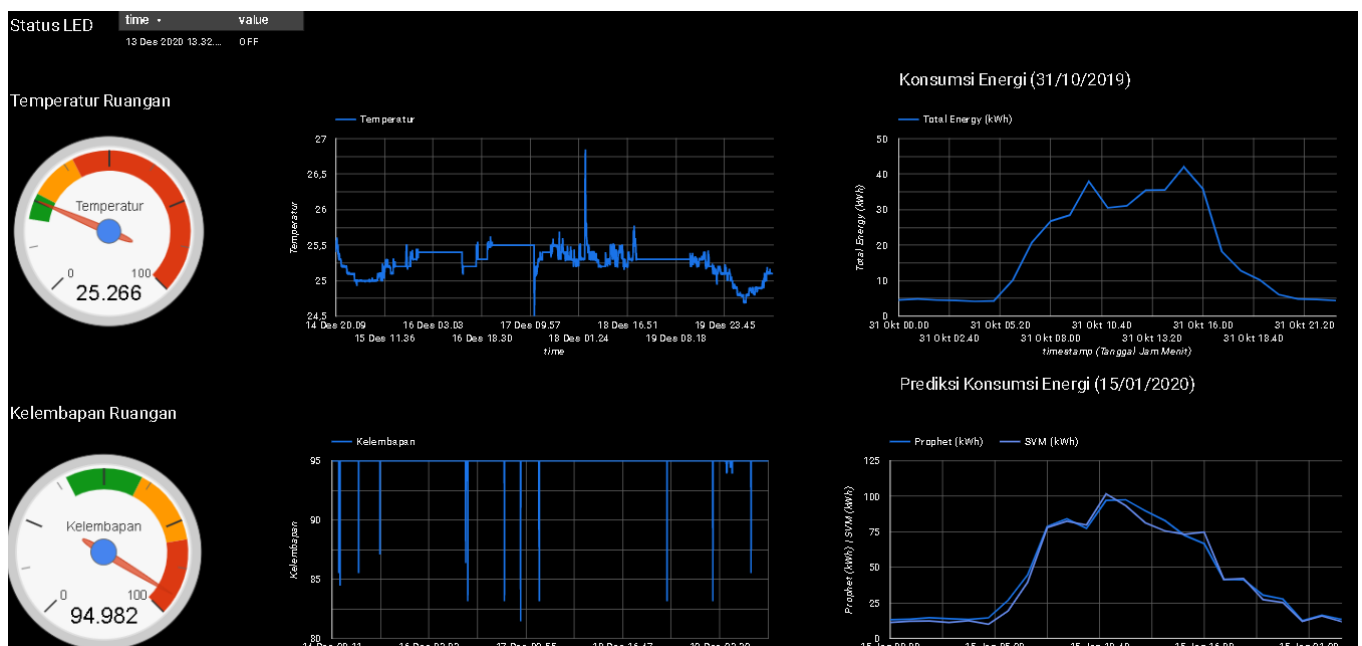
Gambar 6. MQTT broker dan *end-device thing* pada openHAB



**Gambar 7.** Tampilan data dari sensor DHT11 di HABPanel

Pada **Gambar 6** dapat dilihat bahwa baik MQTT broker maupun *end-device* sudah berhasil diintegrasikan pada platform openHAB, dan pada **Gambar 5** dapat dilihat bahwa data dari *end-device* sudah berhasil ditampilkan di antarmuka sederhana pengguna openHAB. Untuk memperindah tampilan dan mempermudah interaksi dengan OpenHAB, dapat digunakan HABPanel sebagai *user interface*. Dengan menggunakan HABPanel, pengguna dapat melakukan fungsi kontrol terhadap device dan menambahkan fitur-fitur lainnya yang sudah tersedia

ataupun menambahkan fitur yang dapat dibuat dan ditambahkan sendiri oleh pengguna. Pada **Gambar 7**, Dapat dilihat *user interface* HABPanel yang sudah menampilkan temperatur dan kelembapan yang dibaca oleh sensor DHT11 dalam bentuk angka dan grafik, Tombol LED untuk melakukan kontrol terhadap LED pada ESP8266, serta status yang menampilkan keadaan ruangan. Pada **Gambar 8**, dapat dilihat tampilan data dari sensor DHT11 pada *google data studio*.



**Gambar 8.** Tampilan data dari sensor DHT11 di *google data studio*

## B. Bacnet Simulator



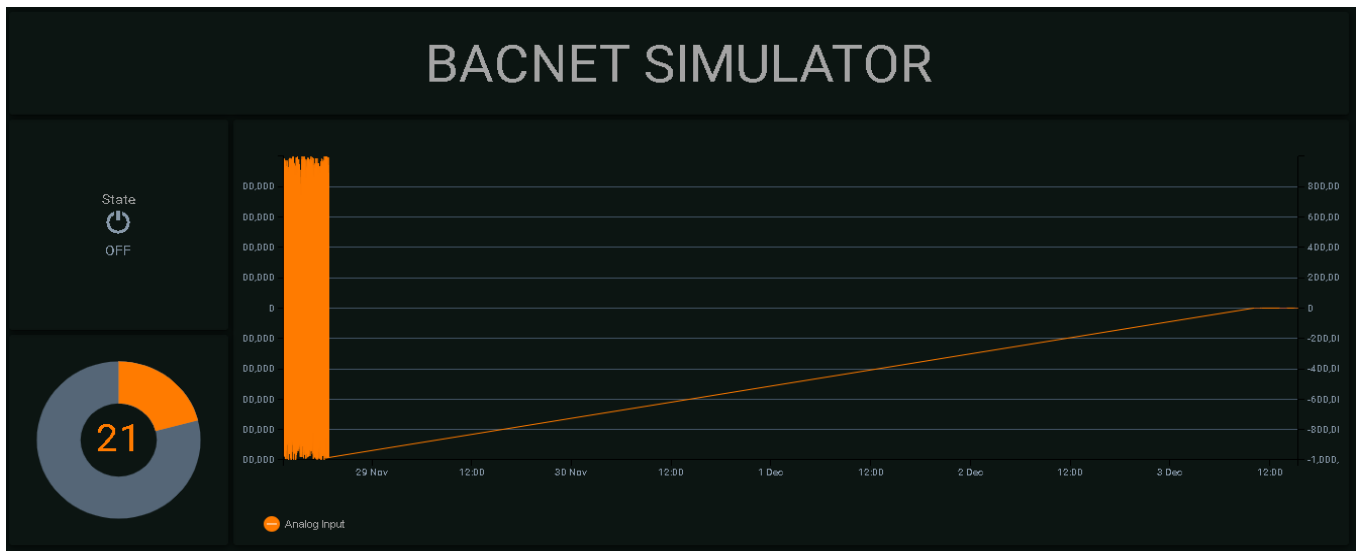
**bacnet\_simulator** ONLINE

Generic MQTT Thing  
mqtt.topic:2361f8c7

**Gambar 10.** Bacnet *simulator thing* pada OpenHAB

bacnet_simulator		
analog_value		20
analog_input		5.8488888740
setting_temp		21
state		<input type="checkbox"/>

**Gambar 9.** Tampilan data dari BACnet *simulator*



**Gambar 11.** Tampilan data dari BACnet *simulator* di HABPanel

Pada **Gambar 9** dapat dilihat bahwa BACnet *simulator* sudah berhasil diintegrasikan pada platform openHAB, dan pada **Gambar 10** dapat dilihat bahwa data dari BACnet *simulator* sudah berhasil ditampilkan di antarmuka sederhana

pengguna openHAB. Pada **Gambar 11**, dapat dilihat tampilan *analog input* dari BACnet *simulator* dalam bentuk grafik, *binary output* dalam bentuk *switch*, dan *analog output* dalam bentuk *slider* untuk melakukan fungsi kontrol.

## C. AC Daikin+Intesis



**BACnet IPv4 network 10.14.30.57** ONLINE

BACnet/IP bridge  
co7io-bacnet:ipv4:10\_14\_30\_255

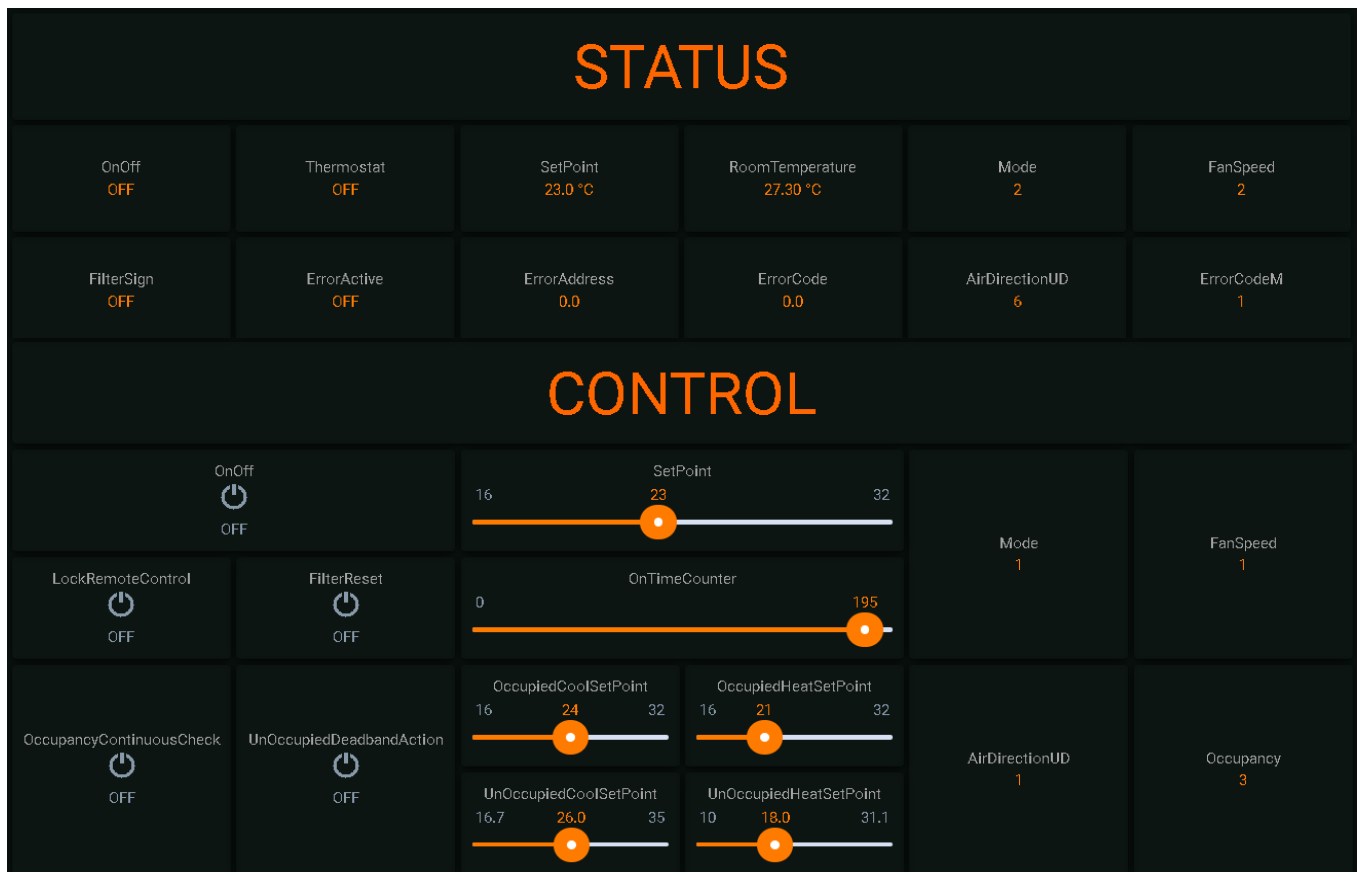


**BACnet/IP device - TF** ONLINE

BACnet/IP device  
co7io-bacnet:ip-device:dc2d6302

**Gambar 12.** Bacnet *bridge* dan *device* (Intesis) *thing* pada OpenHAB





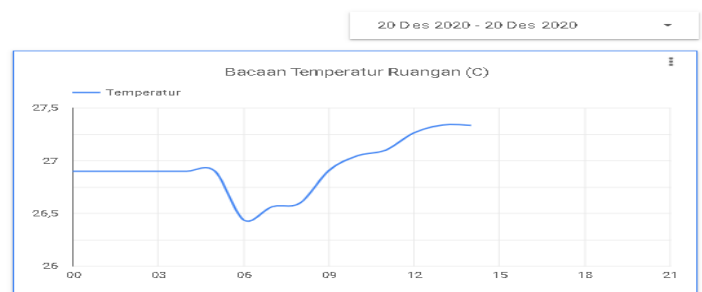
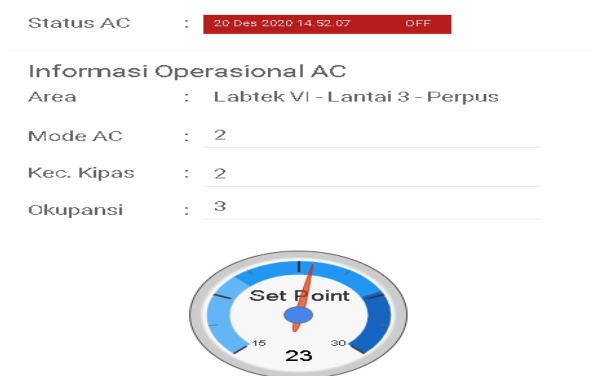
**Gambar 13.** Tampilan data dari Intesis di HABPanel

Pada **Gambar 12** dapat dilihat bahwa Intesis sudah berhasil diintegrasikan pada platform openHAB . Pada **Gambar 13**, dapat dilihat tampilan status dan control dari intesis pada HABPanel. Pada bagian status, ditampilkan data dari *binary input* ( *OnOff*, *Thermostat*, *FilterSign*, *ErrorActive*), *analog input* ( *SetPoint*, *RoomTemperature*, *ErrorAddress*, *ErrorCode*), dan *Multistate input* ( *Mode*, *FanSpeed*, *AirDirection*, *ErrorCode*). Pada bagian control, ditampilkan data *binary output* & *binary value*

(*OnOff*, *LockRemoteControl*, *FilterReset*, *OccupancyContinuousCheck*, *UnoccupiedDeadbandAction*), *analog output&value* ( *SetPoint*, *OnTimeCounter*, *OccupiedCoolSetPoint*, *OccupiedHeatSetPoint*, *UnoccupiedCoolSetPoint*, *UnoccupiedHeatSetPoint*), *multistate output&value* ( *Mode*, *FanSpeed*, *AirDirection*, *Occupancy*) yang dapat di kontrol secara langsung melalui HABPanel. Pada **Gambar 14**, dapat dilihat tampilan data dari intesis di *google data studio*.

## Sistem Otomasi AC

Teknik Fisika - Gedung TP Rachmat ITB



**Gambar 14.** Tampilan data dari Intesis di *google data studio*



## VIII. ANALISIS

### A. Penggunaan Layanan GCP dengan dan tanpa OpenHAB

#### 1) Segi Operasional

Penelitian yang telah dilaksanakan menggunakan dua layanan GCP, yaitu Google Compute Engine dan Google Cloud SQL. OpenHAB dipasang pada mesin virtual di Google Compute Engine. OpenHAB memungkinkan pengguna untuk menghubungkan perangkat, mengaplikasikan protokol komunikasi yang sesuai, dan mengirim kontrol pada perangkat dalam satu *platform* saja. Berbeda jika tidak menggunakan OpenHAB, berdasarkan studi literatur, penulis memerlukan setidaknya lima layanan GCP, yakni Google IoT Core, Google Cloud PubSub, Google Cloud Dataflow, Google Cloud Function, dan Google Cloud SQL.

Koneksi perangkat dengan sistem awan dapat dilakukan dengan Google Cloud IoT Core yang hanya mendukung dua jenis protokol komunikasi, yakni MQTT dan HTTP saja. [9] Keterbatasan ini tentu cukup merepotkan apabila kita akan menghubungkan perangkat yang bekerja dengan protokol selain itu. Pada penelitian ini, contohnya adalah perangkat (3) yang bertukar data menggunakan protokol BACnet. Jika perangkat (3) akan dihubungkan ke GCP tanpa OpenHAB, maka diperlukan sebuah *gateway* yang mengubah BACnet menjadi MQTT agar dapat terhubung. Dalam OpenHAB, tersedia berbagai jenis protokol yang didukung. Pengguna cukup memasang *binding* untuk protokol yang sesuai dan perangkat dapat diregistrasi.

Tanpa OpenHAB, pengguna masih memungkinkan untuk memberikan aksi kontrol melalui Google Cloud Function. Contoh penerapannya terlihat pada **Gambar 15**. [10] Pada diagram tersebut ditampilkan skema kontrol putaran kipas langit-langit. Pertama, sensor membaca nilai temperatur di ruangan. Data tersebut dikirimkan ke layanan Google IoT Core, lalu diteruskan ke layanan Google Cloud Function melalui Google Cloud PubSub. Algoritma yang diimplementasikan pada Google Cloud Platform akan dieksekusi dan mengirimkan kembali perintah ke Google IoT Core. Kemudian, Google IoT Core mengirimkan sinyal kontrol ke

aktuator. Dalam kasus ini adalah kipas langit-langit.



**Gambar 15.** Kasus: kontrol putaran kipas AC [10]

Jika diinginkan sebuah antarmuka terpadu, maka data yang dikumpulkan oleh Google Cloud IoT Core harus disimpan ke media penyimpanan. Hal tersebut dilakukan dengan cara memasang Google Cloud Dataflow setelah Google Cloud PubSub, lalu setelah itu dipasang pula media penyimpanan yang diperlukan, misal Google Cloud SQL. Setelah data disimpan, maka antarmuka dapat dibuat pada Google Data Studio.

Jika dirangkum, mengimplementasikan otomasi rumah pada GCP tanpa menggunakan OpenHAB cukup rumit karena beberapa hal berikut:

1. Layanan GCP yang diperlukan lebih banyak.
2. Masing-masing layanan perlu dikonfigurasi secara terpisah agar dapat bekerja dengan baik. Berbeda dengan OpenHAB, di mana semua dapat saling terhubung dan diperlukan lebih sedikit proses pemrograman dibandingkan tanpa OpenHAB.
3. Keterbatasan protokol yang didukung pada Google Cloud IoT Core menyebabkan penambahan perangkat keras berupa *gateway* tentu menambah kompleksitas sistem otomasi dan menambah biaya.

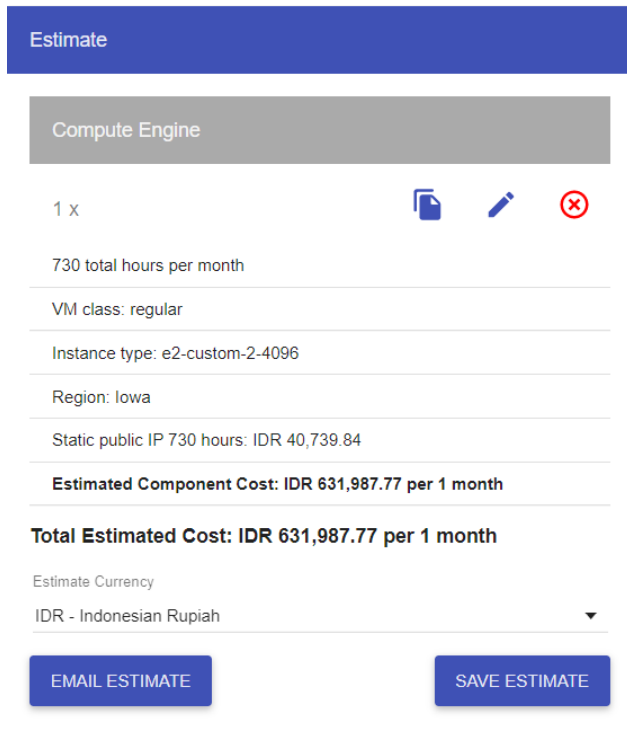
#### 2) Segi Biaya

Selain dari sisi operasional, penambahan layanan GCP saat mengimplementasikan otomasi rumah tentu berpengaruh pula pada biaya yang harus dikeluarkan tiap bulan. Berikut ini dikalkulasi perbandingan biaya yang diperlukan untuk kedua skenario.

Proses kalkulasi berikut menggunakan jumlah data yang dikirim perangkat (1) sebagai demonstrasi. Biaya Google Cloud SQL tidak dihitung karena kedua skenario sama-sama menggunakan layanan tersebut. Kalkulasi dilakukan menggunakan aplikasi kalkulator biaya milik Google Cloud Platform yang dapat diakses melalui [Google Cloud Platform Pricing Calculator](#)

a) Implementasi sistem otomasi dengan OpenHAB

Taksiran harga berikut sesuai dengan spesifikasi mesin virtual yang telah terpasang.



The screenshot shows the Google Cloud Compute Engine pricing interface. At the top is a blue 'Estimate' header. Below it is a grey 'Compute Engine' section header. The configuration includes: 1 instance, 730 total hours per month, VM class: regular, Instance type: e2-custom-2-4096, Region: Iowa, and Static public IP 730 hours: IDR 40,739.84. The 'Estimated Component Cost' is IDR 631,987.77 per 1 month. The 'Total Estimated Cost' is also IDR 631,987.77 per 1 month. At the bottom, there are buttons for 'EMAIL ESTIMATE' and 'SAVE ESTIMATE', and a dropdown for 'Estimate Currency' set to 'IDR - Indonesian Rupiah'.

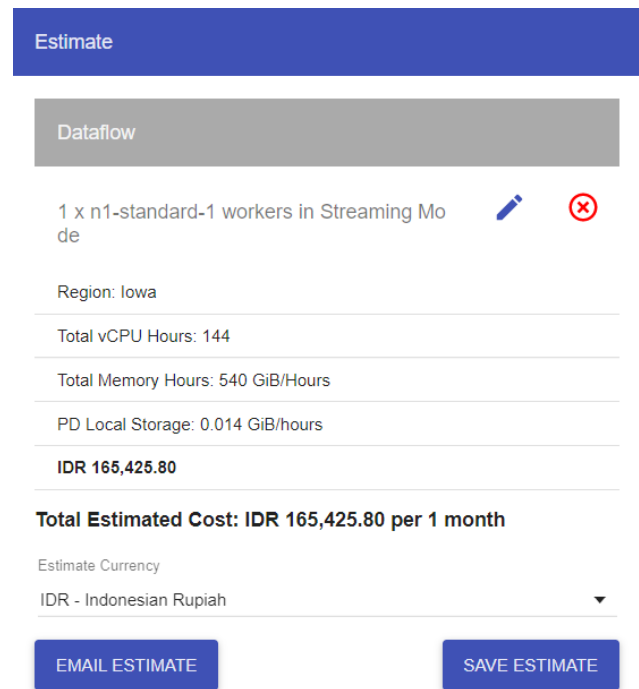
**Gambar 16.** Estimasi biaya Google Compute Engine

Dengan demikian, estimasi biaya total untuk layanan Google Compute Engine untuk OpenHAB adalah sebesar Rp631.987,77/bulan.

b) Implementasi sistem otomasi tanpa OpenHAB

Data yang dikirim oleh perangkat (1) sebesar 4 byte/5 detik, atau sebesar 2,1024 MB/bulan.

- Google Cloud IoT Core  
Google Cloud IoT Core membebaskan biaya layanan jika data yang ditransfer di bawah 250 MB/bulan. Dengan demikian, layanan ini akan gratis untuk perangkat (1).
- Google Cloud Dataflow  
Perangkat (1) mengirim data setiap 5 detik dengan ukuran data 4 byte. Dengan demikian, waktu CPU yang digunakan dalam satu bulan sebesar 144 jam dan digunakan blok sistem pemroses sebesar 100 kiB. Sehingga taksiran harga per bulan adalah sebagai berikut



The screenshot shows the Google Cloud Dataflow pricing interface. At the top is a blue 'Estimate' header. Below it is a grey 'Dataflow' section header. The configuration includes: 1 x n1-standard-1 workers in Streaming Mode, Region: Iowa, Total vCPU Hours: 144, Total Memory Hours: 540 GiB/Hours, and PD Local Storage: 0.014 GiB/hours. The cost is IDR 165,425.80. The 'Total Estimated Cost' is IDR 165,425.80 per 1 month. At the bottom, there are buttons for 'EMAIL ESTIMATE' and 'SAVE ESTIMATE', and a dropdown for 'Estimate Currency' set to 'IDR - Indonesian Rupiah'.

**Gambar 17.** Estimasi biaya Google Cloud Dataflow

- Google Cloud PubSub  
Google Cloud PubSub membebaskan biaya layanan jika data yang ditransfer di bawah 10 GB/bulan. Dengan demikian, layanan ini akan gratis untuk perangkat (1).
- Google Cloud Function  
Google Cloud Function menggratiskan biaya layanannya untuk jumlah eksekusi di bawah dua juta kali per bulan. Pada kasus ini diasumsikan fungsi dieksekusi sebanyak 30 kali untuk mengontrol temperatur ruangan. Dengan demikian, layanan ini akan gratis untuk perangkat (1).

Kalkulasi biaya di atas menunjukkan bahwa penggunaan OpenHAB menjadikan biaya yang harus dikeluarkan per bulan menjadi lebih mahal sebesar 382% jika sistem otomasi rumah tanpa OpenHAB. Namun satu hal yang perlu diperhatikan dari kalkulasi ini adalah perangkat yang digunakan adalah perangkat (1) yang merupakan perangkat yang sangat sederhana dibanding dengan perangkat IoT otomasi rumah yang tersedia di pasaran. Menambah perangkat dan fitur tambahan yang memerlukan transfer data lebih banyak tentu akan membuat biaya berubah lagi. Dengan demikian, pilihan untuk menggunakan OpenHAB atau tanpa OpenHAB harus mempertimbangkan perangkat apa saja yang

terpasang pada sistem sehingga biaya operasional dapat ditekan.

#### B. Antarmuka Pengguna dengan HABPanel dan Google Data Studio

Pada penelitian ini, dibuat dua buah antarmuka pengguna, yaitu pada HABPanel dan Google Data Studio. HABPanel merupakan salah satu fitur bawaan OpenHAB yang berfungsi untuk memvisualisasikan data dan memasukkan kontrol kepada aktuator. Komunikasi yang terjadi antara perangkat dan pengguna bersifat dua arah. Meskipun demikian, HABPanel tidak dapat memvisualisasikan data yang tidak berasal dari perangkat yang terdaftar dalam OpenHAB. Contohnya adalah konsumsi energi listrik dari *smart meter* beserta prediksinya. Selain itu, tampilan grafik seri waktu pada HABPanel memiliki rentang waktu penyajian data yang tetap. Jika rentang tersebut akan diubah, maka keseluruhan dasbor harus dijeda dan diatur pada *widget* terkait.

Berbeda dengan HABPanel, Google Data Studio berfungsi hanya untuk memvisualisasikan data. Dengan demikian, komunikasi hanya berlangsung satu arah yakni dari perangkat kepada pengguna. Google Data Studio mampu mengombinasikan data yang berasal dari luar perangkat yang terdaftar pada OpenHAB. Sebagai contoh telah disebutkan pada paragraf sebelumnya. Konfigurasi grafik seri waktu dapat diatur oleh pengguna dalam mode “view” karena Google Data Studio menyediakan fitur kontrol rentang tanggal untuk fleksibilitas visualisasi data. Meskipun demikian, Google Data Studio tidak dapat menampilkan status data terakhir dan data yang bertipe *string*. Hal ini dapat dilihat pada **Gambar 14** di mana status AC (ON/OFF) adalah data bertipe *string* dan mode AC, kecepatan kipas, serta status okupansi adalah data terakhir. Untuk menampilkan keempat data tersebut, harus digunakan diagram berupa tabel dan dibatasi pengambilan datanya untuk satu data paling baru saja. Hal ini penting dilakukan agar Google Data Studio tidak memasukkan nilai kumulatif maupun rata-rata sebagai status data terakhir.

#### IX. KESIMPULAN

Penggunaan sistem otomasi rumah dapat meningkatkan kenyamanan, keamanan, dan produktivitas penghuni. Perangkat cerdas yang terpasang pada sistem otomasi rumah dapat

dikendalikan melalui *platform* otomasi rumah. Pada penelitian ini, diimplementasikan *platform* otomasi rumah openHAB pada layanan *cloud* menggunakan GCP. Implementasi pada layanan *cloud* memungkinkan penghuni untuk tetap mengakses openHAB kapan saja tanpa khawatir memelihara perangkat keras.

Pada penelitian ini ditemukan bahwa implementasi sistem otomasi rumah dengan atau tanpa *platform* OpenHAB bergantung pada seberapa banyak perangkat yang akan dipantau dan dikendalikan. Jika sistem melibatkan banyak perangkat dan protokol komunikasi, maka memasang OpenHAB pada layanan mesin virtual di Google Compute Engine akan jauh lebih menguntungkan daripada mengintegrasikannya melalui layanan GCP secara satu per satu. Sebaliknya, jika sistem cukup sederhana, maka integrasi dengan layanan GCP secara terpisah dapat menekan biaya operasional sejauh 382%.

Pilihan untuk menggunakan HABPanel atau Google Data Studio pun juga demikian. Perlu ditentukan informasi apa yang ingin digali dari data yang dikirim oleh perangkat. Apabila pengguna menginginkan untuk memantau status terkini dan mengirimkan aksi kontrol, maka HABPanel lebih cocok digunakan daripada Google Data Studio. Namun jika pengguna menginginkan fitur analisis data mendalam, Google Data Studio dapat pula dijadikan sebagai pilihan.

#### DAFTAR PUSTAKA

- [1] B. K. Sovacool and D. D. Furszyfer Del Rio, “Smart home technologies in Europe: A critical review of concepts, benefits, risks and policies,” *Renew. Sustain. Energy Rev.*, vol. 120, no. May 2019, p. 109663, 2020, doi: 10.1016/j.rser.2019.109663.
- [2] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, “A review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT),” in *2015 Internet Technologies and Applications, ITA 2015 - Proceedings of the 6th International Conference*, Nov. 2015, pp. 219–224, doi: 10.1109/ITechA.2015.7317398.
- [3] D. L. Voita, “Home automation system,” no. 19, 1997.

- [4] A. Ben David, "Intelligent home automation," 2010.
- [5] J. Bugeja, A. Jacobsson, and P. Davidsson, "On privacy and security challenges in smart connected homes," in *Proceedings - 2016 European Intelligence and Security Informatics Conference, EISIC 2016*, Mar. 2017, pp. 172–175, doi: 10.1109/EISIC.2016.044.
- [6] A. J. B. Brush, B. Lee, R. Mahajan, S. Agarwal, S. Saroiu, and C. Dixon, "Home automation in the wild," p. 2115, 2011, doi: 10.1145/1978942.1979249.
- [7] D. Tejani, A. M. A. H. Al-Kuwari, and V. Potdar, "Energy conservation in a smart home," in *IEEE International Conference on Digital Ecosystems and Technologies*, 2011, pp. 241–246, doi: 10.1109/DEST.2011.5936632.
- [8] Y. Strengers, J. Kennedy, P. Arcari, L. Nicholls, and M. Gregg, "Protection, Productivity and Pleasure in the Smart Home," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*, May 2019, vol. 645, pp. 1–13, doi: 10.1145/3290605.3300875.

DAFTAR PERTANYAAN DAN JAWABAN PRESENTASI 1

No.	Pertanyaan	Jawaban
1	<p><b>Kelompok 5 (Husnul dan Rizal)</b></p> <p>Ada sebuah arsitektur yang menggambarkan sensor terhubung ke MQTT, untuk bagian itu, prosedurnya bagaimana ya? Apakah sudah didukung sensor atau melalui pengontrol tertentu?</p>	<p>Sensor dipasang pada pengontrol, kemudian data dikirim ke openHAB menggunakan protokol MQTT.</p>
2	<p><b>Kelompok 1 (Hilda dan Iqbal)</b></p> <p>Berarti penyimpanan <i>raw data</i> dan analisis data dilakukan di BigQuery ya?</p>	<p>Penyimpanan <i>raw data</i> dan analisis dilakukan di BigQuery, kemudian data tersebut divisualisasikan di Google Data Studio. Terdapat kemungkinan juga menggunakan Google Machine Learning untuk analisis data karena pada openHAB belum tersedia fitur analisis data.</p>
3	<p><b>Kelompok 6 (Taufiq dan Ilman)</b></p> <p>Jika pertimbangan latensi rendah, bukankah lebih baik menggunakan Google Cloud Bigtable dibanding Google BigQuery?</p>	<p>Tujuan utama adalah dapat melakukan <i>streaming</i> data ke GCP, namun saat ini belum ada pilihan yang pasti.</p>
4	<p><b>Kelompok 1 (Hilda dan Iqbal)</b></p> <p>Pada Google Cloud Storage , terdapat Cloud Spanner, bisa dijelaskan Cloud Spanner tersebut seperti apa? dan apa bedanya dengan Cloud Storage yg lainnya?</p>	<p>Perbedaan terletak pada kapasitas penyimpanannya. Jika data yang diolah sangat besar, maka Cloud Spanner dapat menjadi salah satu opsi.</p>
5	<p><b>Kelompok 9 (Rainda dan Trendy)</b></p> <p>Pada aplikasi IoT untuk otomasi rumah, adakah contoh skenario pengolahan data yang dapat dilakukan di <i>Edge</i> sebelum diolah lebih lanjut di <i>Cloud</i>?</p>	<p>Pengolahan data secara keseluruhan dilakukan di GCP. Saat ini belum ada kasus data diolah secara <i>edge</i> terlebih dahulu karena sistem otomasi rumah memiliki jumlah data yang relatif kecil, berbeda dengan kasus di industri.</p>
6	<p><b>Kelompok 8 (Baninda dan Fikri)</b></p> <p>Pada rencana arsitektur Google Cloud Platform yang ingin dibangun, apakah tidak dibutuhkan fitur Dataprep untuk memfilter data sampling dari perangkat (sensor)?</p> <p>Tanggapan:</p> <p>Analisis pembelajaran mesin sebenarnya dapat dilakukan terlebih dahulu di BigQuery, setelah itu dapat ditampilkan di Google Data Studio</p>	<p><i>Resource</i> yang akan digunakan dapat diubah jika memang masih ada yang diperlukan. Penggunaan Dataprep akan dipelajari dulu sebagai filter sebelum masuk ke BigQuery.</p>

7	<b>Kelompok 8 (Baninda dan Fikri)</b>  Jika dipasang secara <i>on-premise</i> , <i>platform</i> home-automation juga berperan sebagai <i>manager traffic data</i> , <i>device</i> , <i>broker</i> , dan lain-lain. Sementara kalau di GCP kan fungsi-fungsi tersebut sudah disediakan GCP. Jadi apa fungsi utama platform tersebut jika diintegrasikan dengan GCP?	openHAB tidak menyediakan alat untuk analisis dan pengolahan data dengan bantuan pembelajaran mesin. OpenHAB digunakan sebagai pengganti Google IoTCore. openHAB dan GCP merupakan dua hal yang saling independen, namun dapat digabungkan untuk fungsi manajemen yang lebih sederhana. Pada penelitian ini, openHAB bertindak sebagai broker.
8	<b>Kelompok 2 (Heni dan Yusiran)</b>  Dengan menggunakan OpenHAB dapatkah kita mendeteksi pergerakan/penyusupan pada skala gedung secara <i>real time</i> ?	openHAB merupakan penghubung antara pengguna dan perangkat yang terpasang. Fitur apa saja yang disediakan tergantung pada kapasitas perangkatnya. Jika diinginkan, openHAB bisa melakukannya dan mengirimkan datanya secara <i>real time</i> ke antarmuka pengguna.
9	<b>Kelompok 9 (Rainda dan Trendy)</b>  Bahasa pemrograman apa yang akan digunakan?	Pemrograman arduino menggunakan bahasa C, sedangkan openHAB menggunakan Java.

#### DAFTAR PERTANYAAN DAN JAWABAN PRESENTASI 2

No.	Pertanyaan	Jawaban
1	<b>Kelompok 5 (Husnul dan Rizal)</b>  Untuk sampling rate sensor berapa? pengiriman nya bagaimana? Realtime atau menyimpan (cloud SQL)? Bagaimana menghitung dan mengecek bandwidth ke cloud?	SR 5 detik ke VM;; iya realtime; kami tdk mempertimbangkan bandwidth dan tidak menghitung juga, karena percobaan kemarin sukses.
2	<b>Kelompok 9 (Rainda dan Trendy)</b>  Hasil sementara grafik lurus tgl 8-9, apakah di sini sensornya ada kendala atau tidak melakukan sampling?	Memang sensornya membaca nilai kelembapan sekian, jika pun ada masalah mungkin di sensitifitas sensornya. Kalau Grafik yang turun drastis karena tdk ada data (mati).
3	<b>Pak Justin</b>  Pak Justin: Apakah sudah pernah di explore sampai sejauh mana kemampuan GDS? Apakah bisa user interaktif atau bahkan memberi command? Apakah tampilan bisa diubah? Apakah ada fitur untuk event atau warning?	GDS hanya menyediakan reporting, interaktif belum dicoba dari sisi pengguna namun dari sisi editor dapat disesuaikan. Tampilan juga dapat disesuaikan. Untuk event belum tahu.

DAFTAR PERTANYAAN DAN JAWABAN PRESENTASI 3

No.	Pertanyaan	Jawaban
1	<p><b>Kelompok 5 (Husnul dan Rizal)</b></p> <p>Ingin bertanya, untuk OpenHAB/DataStudio apakah ada fitur/cara utk preprocessing data sebelum ditampilkan ke widget?</p>	<p>Kalau untuk preprocessing data secara sederhana ada, seperti jumla desimal, satuan,dll. Tapi untuk preprocessing yang lebih kompleks seperti dataproc tidak bisa</p>
2	<p><b>Kelompok 9 (Rainda dan Trendy)</b></p> <p>Alarm tersebut akan berbunyi ketika pada kondisi apa ya???</p>	<p>Alarm pada proyek yang dikerjakan akan berbunyi jika LED menyala, yakni ketika tombol untuk menyalakan LED ditekan.</p>
3	<p><b>Kelompok 6 (Taufiq dan Ilman)</b></p> <p>izin bertanya untuk dashboard dan device mengapa pertimbangan komunikasi dua arah, apakah ada kendala? sama mau nanya Threshold alarm suhu berapa atau kelembapan berapa?</p>	<p>Komunikasi 2 arah justru adalah hal yang baik, bukan dilakukan karena ada kendala, tetapi agar data bisa mengalir secara 2 arah, dimana data dari device dapat diterima dashboard untuk ditampilkan, dan data dari dashboard dapat diterima device untuk melakukan fungsi kontrol.</p>
4	<p><b>Kelompok 7 (Mukhlas dan Paul)</b></p> <p>1) Ketika berbicara tentang pricing apakah sudah pernah dicoba untuk dihitung perkiraan pricing gcp-nya? Karena setau saya harga untuk VM GCP itu cukup mahal jika dibandingkan dengan layanan IoT Core/PubSub/dll yang murah dan ada kuota gratis per bulannya.</p>	<p>Untuk proyek secara kecil, penggunaan VM untuk IoT memang lebih mahal. Namun, untuk proyek skala besar yang memiliki debit data yang besar, penggunaan IoT Core,Pubsub,Function/Dataflow akan menjadi lebih mahal dibandingkan dengan biaya penggunaan VM</p>
5	<p><b>Kelompok 8 (Baninda dan Fikri)</b></p> <p>Apakah dashboard di openhab bisa diakses via internet?</p>	<p>Bisa. Karena OpenHAB diimplementasikan pada Cloud, maka bisa diakses kapan saja dan dimana saja asalkan port firewall sudah dibuka, ataupun bisa tanpa firewall yaitu menggunakan OpenHAB cloud connector</p>
6	<p><b>Kelompok 5 (Husnul dan Rizal)</b></p> <p>dimana saja openhub bisa di install?</p>	<p>OpenHAB bisa di install di single board computer seperti Raspberry Pi dan komputer baik secara lokal ataupun pada cloud asalkan memenuhi spesifikasi yang dibutuhkan.</p>
7	<p><b>Kelompok 9 (Rainda dan Trendy)</b></p> <p>apakah bisa setting kontrol LEDnya secara otomatis?</p>	<p>Bisa. Hal tersebut dapat dilakukan dengan mengkonfigurasi file rules pada OpenHAB untuk melakukan proses Otomasi lebih lanjut.</p>