Implementasi Arsitektur *Serverless Internet of Things* pada *Monitoring Cold Chain*

Aisyah Mulyani[1], Unan Yusmaniar Oktiawati [1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
aisyahmulyani@mail.ugm.ac.id
*Korespondensi: unan_yusmaniar@ugm.ac.id

Abstract – With the current technological developments, the need for monitoring in cold chain logistics as well as the problem of limitations on IoT devices such as data storage and computing capabilities can be handled with the help of cloud technology. Cloud has a concept called serverless. In this study, researchers tried to examine serverless services that can be integrated with IoT devices. The IoT devices used consist of nodes and gateways. In this final project research, testing and analysis of gateway devices and serverless architecture are carried out. NodeMCU which is part of the gateway device and AWS IoT as a serverless service will be integrated in this research. The serverless architecture system that was built also has a function to store data and send notifications to the user. Meanwhile, data monitoring will be carried out through Grafana. The results of AWS IoT performance testing when integrated with IoT gateway devices through QoS testing according to the TIPHON standard are categorized as satisfactory and the implemented system functions as expected.

Keywords : Internet of Things, Cloud Computing, Serverless, Amazon Web Services

Intisari – Dengan adanya perkembangan teknologi saat ini, kebutuhan *monitoring* pada *cold chain logistic* serta masalah keterbatasan pada perangkat IoT seperti kemampuan penyimpanan data dan komputasi dapat ditangani dengan bantuan teknologi *cloud*. *Cloud* memiliki konsep yang dinamakan dengan *serverless*. Pada penelitian ini peneliti mencoba untuk meneliti layanan *serverless* yang dapat diintegrasikan dengan perangkat IoT. Perangkat IoT yang digunakan terdiri dari node dan gateway. Pada penelitian proyek akhir ini dilakukan pengujian dan analis dari perangkat gateway dan arsitektur *serverless*. NodeMCU yang merupakan bagian dari perangkat gateway dan AWS IoT sebagai layanan *serverless* akan diintegrasikan pada penelitian ini. Sistem arsitektur *serverless* yang dibangun juga memiliki fungsi untuk menyimpan data dan mengirimkan notifikasi kepada user. Sedangkan *monitoring* data akan dilakukan melalui Grafana. Hasil pengujian performa AWS IoT saat diintegrasikan dengan perangkat IoT gateway melalui pengujian QoS menurut standar TIPHON berkategorikan memuaskan dan sistem yang telah diimplementasikan berfungsi sesuai yang diharapkan.

Kata kunci : ***Internet of Things, Cloud Computing, Serverless, Amazon Web Services***

I. PENDAHULUAN

Pertumbuhan penduduk yang meningkat menyebabkan meningkatkan kebutuhan makanan. Salah satu produk makanan yang membutuhkan jaminan mutu yang baik adalah produk *perishable* yaitu produk yang rentan busuk atau rusak seperti buah – buahan, sayuran, daging dan ikan. Dalam pendistribusian produk *perishable* atau yang biasa disebut dengan *cold chain* logistic, terdapat dua kegiatan yaitu kegiatan logistik dan pengendalian serta pemantauan suhu. Suhu dari produk *perishable* perlu dijaga dengan baik agar kualitas makanan tetap terjamin. Salah satu cara untuk memastikan suhu tetap rendah adalah dengan melakukan *monitoring* suhu.

Pada penelitian ini dibuat sebuah arsitektur *cloud* yang berfungsi untuk diintegrasikan dengan perangkat IoT sehingga dapat melakukan pemantauan atau *monitoring* suhu secara *real-time*. Perangkat tersebut terdiri dari node dan gateway. Node akan berfungsi sebagai alat pengukur suhu yang diletakkan disetiap *box* pengiriman sedangkan gateway berfungsi sebagai router yang akan mendistribusikan data suhu ke *cloud* untuk dapat di*monitoring* oleh pengguna dimanapun berada. Oleh karena itu pembuatan arsitektur *cloud* untuk dapat menerima, menyimpan dan serta mengirimkan notifikasi mengenai kondisi tertentu pada *box* pengiriman sangat dibutuhkan.

Dalam mengintegrasikan perangkat IoT dan *cloud*, penelitian ini akan memanfaatkan layanan arsitektur *serverless* pada Amazon Web Services yaitu AWS IoT. AWS IoT akan berperan sebagai penghubung antara perangkat dan arsitektur *cloud*. AWS IoT akan menerima data dan melanjutkan data tersebut untuk disimpan maupun diolah untuk fungsi lainnya seperti notifikasi dan pemantauan.

Selain AWS IoT, layanan *serverless* yang akan digunakan pada penelitian ini adalah AWS Lambda. Fungsi dari Lambda pada penelitian ini adalah melakukan *query* ke *database* setiap beberapa menit lalu mengirimkan notifikasi melalui AWS SNS (*Simple Notification Service*) ketika terjadi kondisi tidak normal pada suhu.

Berdasarkan penjelasan di atas, penulis akan mencoba untuk melakukan penelitian mengenai performa AWS IoT sebagai layanan *serverless* yang di integrasikan dengan perangkat IoT gateway serta bagaimana layanan *serverless* tersebut dapat menerima, menyimpan dan menampilkan data secara real-time serta mengirimkan notifikasi ke pengguna. Serta untuk memudahkan pemantauan suhu, pada penelitian ini akan mengimplementasikan penggunaan Grafana untuk memvisualisasikan data yang telah disimpan pada *database* di AWS Timestream.

II. TEORI PENDUKUNG

A. *Cold chain*

Cold chain adalah rantai pengiriman yang terdiri dari serangkaian aktivitas penyimpanan dan distribusi yang tidak terputus dengan mempertahankan kisaran suhu yang ditentukan. Salah satu cara untuk memastikan pendistribusian produk *perishable* pada *cold chain logistic* berjalan dengan baik, dilakukan pemantauan besaran suhu. Pemantauan besaran suhu dengan menggunakan cara manual

adalah dengan melakukan pengecekan ke dalam *tiap cold storage* secara berkala. Kekurangannya adalah menghabiskan banyak waktu terutama apabila terdapat banyak *cold storage* dalam satu pengiriman. Oleh karena itu diperlukan sebuah penerapan teknologi yang mampu membangun sebuah sistem *monitoring cold chain* yang lebih efisien dan real-time, yakni pemanfaatan *Internet of Things* (IoT).

B. *Internet of Things*

Internet of Things (IoT) adalah sebuah konsep di mana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan tanpa memerlukan adanya interaksi dari manusia ke manusia atau dari manusia ke komputer. Internet of Things (IoT) adalah struktur di mana objek, orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke komputer [1].

Salah satu pemanfaaatan IoT yang berhubungan dengan sistem monitoring *cold chain* telah dilakukan oleh [2] mengenai "*On the Application of Internet of Things (IOT) in Cold chain Logistics Management*". Penelitian tersebut menjelaskan mengenai manfaat penerapan IoT dalam *cold chain logistics* seperti meminimalisir *error*, melindungi asset dan kargo, serta menjamin kualitas dari produk *perishable*.

C. *Serverless*

Severless Architecture atau arsitektur tanpa *server* merupakan langkah untuk membangun dan menjalankan berbagai aplikasi dan layanan tanpa perlu mengelola infrastruktur. *Server* masih menjalankan suatu aplikasi, tetapi semua manajemen *server* dilakukan oleh penyedia layanan. Sehingga tidak lagi perlu menyediakan atau mengelola *server* untuk menjalankan aplikasi, *database*, dan sistem penyimpanan. Dengan komputasi tanpa *server*, tugas manajemen infrastruktur seperti penyediaan kapasitas dan patching ditangani oleh penyedia layanan, sehingga hanya dapat fokus pada penulisan kode yang melayani pelanggan. Dengan demikian hal ini dapat mengurangi biaya pengeluaran tambahan dan juga dapat menghemat waktu dan energi bagi *developer* [3].

Arsitektur *serverless* bukan menandakan bahwa pada arsitektur tersebut pengguna tidak menggunakan *server* sama sekali. Yang dimaksud dengan *serverless* dalam hal ini adalah dapat berkurangnya interaksi pengguna dalam melakukan *managing* atau *deploying server*. Apabila sebelumnya pengguna perlu melakukan SSH kedalam untuk mengakses *server* maka dalam konsep *serverless* pengguna dapat melakukan hal tersebut melalui konsol web. Selain memiliki sifat minimal *deployment* atau *simple*, arsitektur *serverless* memiliki beberapa sifat atau kelebihan seperti *auto-scaling, cost saving dan time saving.*

D. AWS IoT

AWS IoT adalah layanan *cloud* terkelola yang memungkinkan perangkat yang terhubung dengan mudah dan aman berinteraksi dengan aplikasi *cloud* dan perangkat lain. AWS IoT mendukung protokol HTTP, WebSockets, dan MQTT, protokol komunikasi ringan yang dirancang khusus untuk mentolerir koneksi terputus-putus, meminimalkan jejak kode pada perangkat, dan mengurangi kebutuhan bandwidth jaringan. Karena mendukung protokol MQTT, AWS IoT dapat berperan sebagai broker dan melakukan subscribe dan publish. AWS IoT juga dapat dikolaborasikan dengan berbagai layanan lainnya pada AWS seperti SNS, *database*, IoT Analytics dan lain – lain. Selain itu AWS IoT juga menyediakan *Certificate Authority* (CA) sehingga hanya perangkat yang yang diberikan otorisasi saja yang dapat terhubung dengan AWS IoT.

E. Protokol Komunikasi MQTT

Protokol komunikasi merupakan seperangkat aturan yang digunakan untuk melakukan komunikasi agar dua belah pihak yang berkomunikasi dapat saling memahami. Protokol komunikasi IoT dibagi menjadi dua yaitu *IoT data protocol* dan *Network protocol for IoT*. Dalam penelitian ini akan dibahas mengenai *IoT data protocol* yaitu protokol yang digunakan perangkat IoT untuk berkomunikasi dengan pengguna melalui internet. Jenis *IoT data protocol* yang digunakan dalam penelitian ini adalah MQTT.

Message Queue Telemetry Transport (MQTT) adalah sebuah protokol komunikasi data *machine to machine* (M2M) yang berada pada layer aplikasi, MQTT bersifat *lightweight message* artinya MQTT berkomunikasi dengan mengirimkan data pesan yang memiliki header berukuran kecil yaitu hanya sebesar 2 bytes untuk setiap jenis data, sehingga dapat bekerja di dalam lingkungan yang terbatas sumber dayanya seperti kecilnya bandwidth dan terbatasnya sumber daya listrik, selain itu protokol ini juga menjamin terkiriminya semua pesan walaupun koneksi terputus sementara, protokol MQTT menggunakan metode publish/subscribe untuk metode komunikasinya [4].

F. Grafana

Grafana adalah perangkat lunak open-source yang digunakan untuk mengolah data menjadi informasi yang ditampilkan dalam format grafik. Grafana menyediakan banyak fitur yang powerful dalam melakukan *monitoring* dan analisa sehingga menjadi terkenal dan banyak digunakan di perusahaan sebagai platform *monitoring*. Grafana dapat dijalankan pada sistem operasi Gnu / Linux, Windows dan MacOS. Umumnya grafana mendapatkan metric atau data yang diolah dengan melakukan *query* ke *database*. Grafana kompatibel dengan banyak *database* opensource maupun berbayar seperti MySQL, Influx DB, AWS Timestream dan lain – lain.

G. QoS

QoS didesain untuk membantu *end user (client)* menjadi lebih produktif dengan memastikan bahwa user mendapatkan performansi yang handal dari aplikasi-aplikasi berbasis jaringan. QoS mengacu pada kemampuan jaringan untuk menyediakan layanan yang lebih baik pada trafik jaringan tertentu melalui teknologi yang berbeda-beda [5].

Terdapat beberapa parameter QoS seperti throughput, *delay*, jitter, *packet* loss dan *packet delivery*. Namun dalam penelitian ini hanya akan digunakan 2 parameter yaitu *packet delivery* dan *delay*. *Packet delivery* diartikan sebagai jumlah kedatangan paket yang berhasil sampai ke penerima, menurut standar TIPHON (*Telecommunications and Internet Protocol Harmonization Over Network*) kategori *packet delivery* dibagi ke dalam empat kategori. Tabel 1 menunjukkan indeks kategori *packet delivery*.

Tabel 1. Kategori *Packet delivery*

Kategori	Presentase	Indeks
Sangat Bagus	100%	4
Bagus	97%	3
Sedang	85%	2
Jelek	75%	1

Sedangkan *delay* merupakan waktu penundaan paket saat melakukan proses pengiriman dari satu titik ke titik lain yang dituju. *Delay* menggunakan satuan milidetik (ms). Kategori *delay* menurut standar TIPHON tertera pada Tabel 2 kategori *delay*.

Tabel 2. Kategori *Delay*

Kategori	Nilai *Delay*	Indeks
Sangat Bagus	<150 ms	4
Bagus	150 s/d 300 ms	3
Sedang	300 s/d 450 ms	2
Jelek	>450 ms	1

Hasil dari perhitungan *packet delivery* dan *delay* akan dikalkulasikan sesuai kategori QoS, seperti terletak pada Tabel 3. Kategori QoS Versi TIPHON.

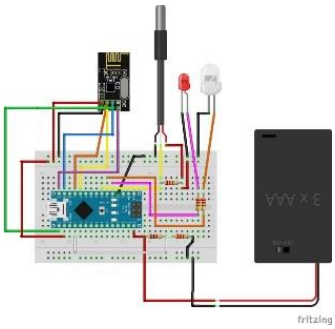
Tabel 3. Kategori QoS

Kategori	Presentase	Indeks
3,8 – 4	95 – 100%	Sangat Memuaskan
3 – 3,75	75 – 95,75%	Memuaskan
2 – 2,99	50 – 74,75%	Kurang Memuaskan
1 – 1,99	25 – 49,75%	Buruk

III. METODOLOGI PENELITIAN

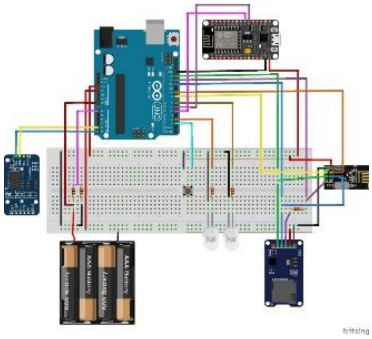
A. Perancangan Perangkat dan Alur Sistem

Perancangan perangkat terdiri dari 2 perancangan yaitu perancangan node dan perancangan gateway. Perancangan node yang terdiri dari 3 komponen utama yaitu Arduino Nano sebagai microcontroller, NRF24L01 sebagai modul komunikasi serta modul DSB18 sebagai sensor suhu sebagaimana disajikan pada Gambar 1 berikut.



Gambar 1. Perancangan Node

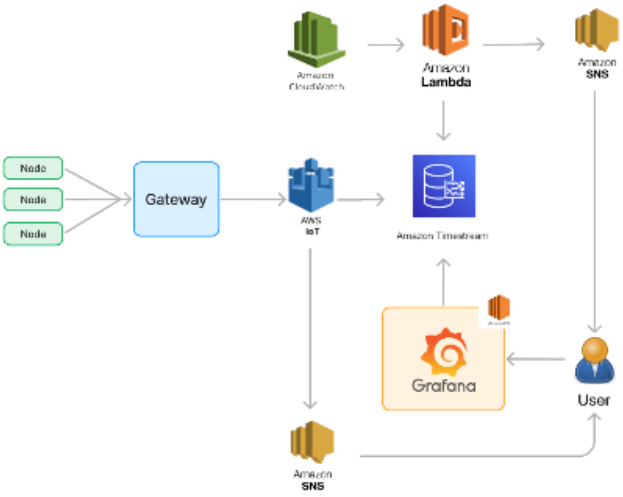
Sedangkan untuk rangkaian gateway, perangkat terdiri dari lebih banyak bagian seperti pada Gambar 2 berikut.



Gambar 2. Perancangan Gateway

Gateway terdiri dari dua microcontroller yaitu Arduino UNO dan NodeMCU ESP8266. Gateway juga disertai beberapa modul tambahan untuk menerima data dari node. NodeMCU akan menerima data yang sebelumnya telah dikumpulkan Arduino UNO menggunakan modul NRF24L01. Data dikirimkan melalui pin serial D7 dan D6. Pengiriman data tersebut dilakukan dengan memanfaatkan *library* Arduino Json. Setelah NodeMCU menerima data dari Arduino UNO, data tersebut akan digabungkan ke dalam variable menggunakan fungsi sprint membentuk sebuah pesan JSON untuk dikirimkan menggunakan protokol MQTT. Kemudian data yang diterima oleh AWS IoT Core akan diteruskan ke *database* Timestream.

Untuk *monitoring* data, *database* akan dihubungkan dengan Grafana *server*. Grafana akan menambahkan Timestream *Database* sebagai *data source*, menambahkan info ARN Timestream *Database* yang digunakan ke dalam konfigurasi sehingga Grafana dapat melakukan *query* dan menampilkan visualisasi dari data tersebut. Sedangkan untuk notifikasi, AWS Lambda akan melakukan *query* ke *database* setiap beberapa menit yang dijadwalkan menggunakan AWS CloudWatch EventBridge. Notifikasi akan dikirimkan ke email user yang telah melakukan subscription ke AWS SNS. Gambar 3 berikut akan menunjukkan ilustrasi dari alur transmisi data dan kerja sistem.



Gambar 3. Arsitektur *cloud*

B. Skenario Pengujian

1) Pengujian Performa AWS IoT

Pengujian performa AWS IoT berguna untuk mengetahui performa yang dimiliki AWS IoT sebagai layanan *serverless* untuk pengintegrasian *cloud* dengan perangkat IoT.

a. *Packet delivery*

Pengujian *packet delivery* dilakukan dengan dua tahap. Tahap pertama adalah memastikan tumpukan data yang diterima oleh AWS IoT sesuai dengan yang dikirimkan NodeMCU. Pengujian akan dilakukan secara berulang dengan mencoba mengirimkan informasi node dengan ID terkecil ke ID terbesar. Hal ini merupakan hal yang penting dikarenakan akan berpengaruh pada catatan waktu kapan informasi tersebut disimpan. Setelah informasi 3 node dengan ID 101, 103 dan 104 berhasil diterima dengan urutan yang sesuai tahap selanjutnya adalah melakukan pengujian delivery. Percobaan dilakukan dengan mengirimkan data dari masing – masing node secara bergantian sebanyak 30 pesan.

b. *Delay*

Pengujian *delay* dilakukan dengan 3 kali percobaan dengan masing – masing percobaan mengirimkan 10 data dari salah satu node. Selanjutnya bandingkan waktu pengiriman data pada NodeMCU dan penerimaan data pada AWS IoT.

c. Pengujian Waktu Penyimpanan

Pengujianini dilakukan untuk mengetahui waktu yang dibutuhkan AWS IoT menyimpan data ke *database* Timestream dengan membandingkan waktu penerimaan data pada AWS IoT dengan waktu yang tercatat pada *database*.

2) Pengujian Fungsional

Pengujian fungsionalitas bertujuan menguji keberhasilan sistem dalam menerima, menyimpan dan menampilkan data secara realtime serta mengirimkan notifikasi ke pengguna. Beberapa pengujian fungsional yang dilakukan pada penelitian ini antara lain:

a. AWS IoT *Rules* dapat memicu SNS mengirimkan notifikasi pada saat node terputus

b. CloudWatch dapat mengatur kerja Lambda untuk mendeteksi suhu lebih dari 20°C dan mentrigger SNS mengirimkan notifikasi ke email user yang telah melakukan subscribe.

3) Pengujian *Monitoring* Grafana

Pengujian ini dilakukan untuk menguji apakah implementasi dan rancangan dari sistem *monitoring* yang telah dilakukan berjalan dengan baik, Grafana dapat memvisualisasikan data ke dalam bentuk grafik serta Grafana dapat mengkonversi grafik ke dalam dokumen berformat PDF.

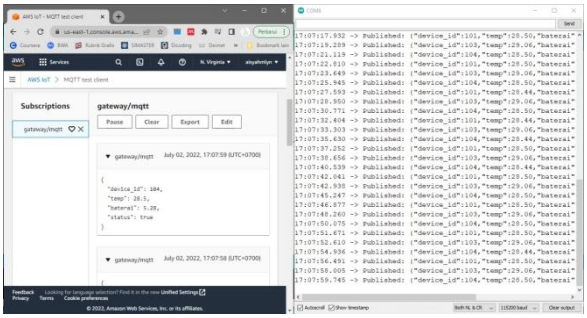
IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian Performa AWS IoT

1) Hasil Pengujian *Packet delivery*

Pada tahap pertama pengujia yaitu urutan data di diterima di AWS IoT. Melalui pengujian ini terbukti bahwa AWS IoT mampu menerima data yang dikirimkan NodeMCU menggunakan protokol MQTT. Urutan data yang diterima AWS IoT sesuai dengan yang dikirimkan perangkat gateway.

Gambar 4 merupakan salah satu bukti hasil pengujian. Pada pengujian tersebut, NodeMCU seperti yang terlihat pada serial monitor berhasil meneruskan pesan atau informasi node secara urut dengan melakukan published. Kemudian pada halaman AWS IoT, menggunakan MQTT test client, AWS IoT melakukan subscribe terhadap topik yang digunakan, lalu pesan baru akan muncul setiap kali perangkat mengirimkan pesan.



Gambar 4. Pengujian urutan data

Packet delivery merupakan *packet* yang berhasil sampai ke penerima. Penerima dalam hal ini adalah AWS IoT, namun AWS IoT dalam website testingnya hanya dapat menyimpan informasi data sebanyak 10 data. Oleh karena itu jumlah data yang diterima dilihat melalui hasil eksport data di Grafana. Dari hasil eksport data tersebut dapat diketahuni bahwa semua data yang dikirimkan dapat diterima oleh AWS IoT, masing – masing data node yg dikirimkan lengkap sejumlah 30 data.

2) Hasil Pengujian *Delay*

Untuk mendapatkan detail waktu diterimanya pesan pada AWS IoT, data perlu di eksport. Hasil file eksport akan berformat json dimana waktu disimpan ke dalam format timestamp. Format timestamp tersebut kemudian dikonversi ke waktu yang dapat dimengerti manusia. Selanjutnya waktu tersebut dan waktu pengiriman perangkat akan dibandingkan untuk mendapatkan nilai *delay* dan *delay* rata – rata. Tabel 4 menunjukkan hasil pengujian *delay*.

Tabel 4. Hasil Pengujian *Delay*

Pesan ke - n	Pengujian *Delay* Pengiriman data ke – n (ms)		
	1	2	3
1	226	244	469
2	274	251	241
3	263	447	233
4	199	224	283
5	217	213	257
6	219	223	257
7	213	238	273
8	209	263	222
9	234	239	267
10	240	141	233
Delay Rata-rata	229,4	248,3	273,5

Tabel 5. Hasil Pengujian Waktu Penyimpanan Data

Pesan ke - n	Pengujian waktu Penyimpanan data ke – n (ms)		
	1	2	3
1	462	485	485
2	468	471	492
3	452	284	489
4	448	580	469
5	521	428	561
6	471	474	495
7	530	477	480
8	466	478	493
9	455	587	592
10	452	470	503
Waktu Rata-rata	472,5	473,4	505,9

Melalui pengujian ini dapat terlihat bahwa *delay* terbesar selama pengiriman berada pada 469 ms sedangkan *delay* terkecil mencapai 141 ms. Sedangkan nilai *delay* rata – rata untuk ketiga percobaan tersebut adalah 250 ms. Penilaian mengenai hasil pengujian ini apabila mengacu pada standar TIPHON akan berindeks 3 yang berarti berkategori "Bagus".

Hasil pengujian QoS untuk delay dan packet delivery dalam penelitian ini akan dikalkulasikan sesuai standarisasi TIPHON. Seperti yang telah disebutkan sebelumnya indeks untuk pengujian packet delivery adalah 4 sedangkan untuk delay adalah 3. Bila dikalkulasikan nilai Quality of Services dari parameter delay dan packet delivery, maka pengujian QoS dalam pengiriman data dari perangkat gateway NodeMCU menuju AWS IoT bernilai 3,5 atau masuk dalam kategori "Memuaskan".

3) Hasil Pengujian Waktu Penyimpanan Data

Selain menguji *delay* pengiriman data, setelah mengetahui waktu penerimaan AWS IoT, dilakukan pula pengujian proses penyimpanan data (Tabel 5). Dengan pengujian ini, dapat diketahui berapa lama waktu yang dibutuhkan AWS IoT untuk menyimpan data yang diterima ke dalam *database*. Pengujian ini dilakukan dengan membandingkan nilai pada AWS IoT dan yang tercatat pada *database* Timestream. Meskipun terdapat perbedaan waktu dikarenakan region pada AWS yang digunakan berada di North Virginia yang menggunakan format waktu UTC, pengujian ini akan dilakukan dengan membandingkan nilai menit dan detik yang tercatat. Berikut merupakan hasil pengujian penyimpanan data pada AWS IoT.

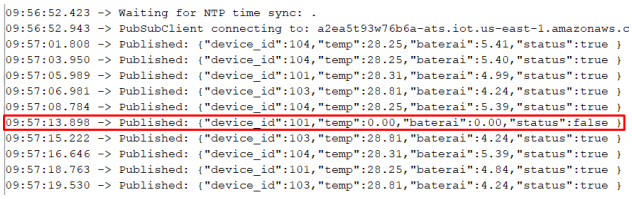
Melalui hasil pengujian ini dapat terlihat bahwa waktu terlama AWS IoT untuk menyimpan data adalah 592 ms sedangkan waktu tersingkatnya yaitu 284 ms. Sedangkan nilai waktu rata – rata untuk ketiga percobaan tersebut adalah 483.9 ms.

B. Hasil Pengujian Fungsional

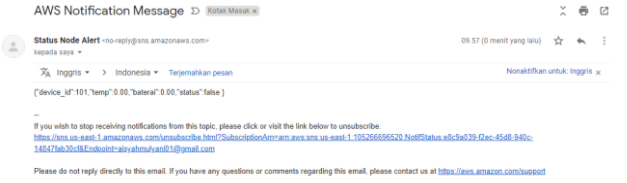
1) Hasil Pengujian AWS IoT *Rules*

Pengujian fungsional yang pertama adalah AWS IoT *Rules*. Setelah pada pengujian bagian sebelumnya AWS IoT telah berhasil menerima data dari perangkat gateway secara urut, pengujian selanjutnya adalah pengujian AWS IoT *Rules* yang telah dikonfigurasi dengan dua fungsi yaitu fungsi penyimpanan dan notifikasi. *Rules* yang dibuat akan membuat perangkat secara tidak langsung terhubung dengan layanan lain seperti *database* untuk penyimpanan dan AWS SNS untuk notifikasi. Pengujian mengenai penyimpanan sebelumnya telah dilakukan maka pada bagian ini akan dibahas mengenai fungsi notifikasi.

Berikut merupakan hasil dari pengujian notifikasi node yang tidak aktif. Node yang tidak aktif tersebut memiliki status *false*. Ketika AWS IoT mendapati pesan dengan kategori tersebut maka AWS IoT akan memicu AWS SNS untuk mengirimkan notifikasi berupa email. Notifikasi yang dikirimkan berupa informasi ID node yang tidak aktif seperti dapat dilihat pada Gambar 5 dan 6.



Gambar 5. Pengujian notifikasi melalui AWS IoT



Gambar 6. Hasil notifikasi yang diterima

2) Hasil Pengujian Lambda dan CloudWatch

Pada penelitian ini Lambda digunakan untuk membuat sebuah fungsi agar dapat melakukan *query* ke *database* untuk melihat rata – rata besar suhu dalam 2, 5 atau 10 menit terakhir dan mengecek apakah nilai tersebut normal atau tidak. Satu fungsi Lambda yang dibuat pada penelitian ini digunakan untuk mengecek satu Node.

Fungsi Lamba berikut ini diatur kerjanya oleh Cloud Watch EventBridge sehingga penjadwalan kerja fungsi Lambda dilakukan di Cloud Watch EventBridge. Pengujian Lambda dan CloudWatch dilakukan untuk membuktikan bahwa fungsi yang telah dijalankan di Lambda dapat berfungsi dengan baik serta membuktikan bahwa CloudWatch mampu mengatur waktu kerja Lambda (Tabel 6).

Tabel 6. Hasil Pengujian Lambda dan CloudWatch

Fungsi yang diuji	Perulangan	Hasil
Dapat mengirimkan notifikasi saat suhu lebih dari 20°C	2 menit	Berhasil
	5 menit	Berhasil
	10 menit	Berhasil

Berikut ini merupakan salah satu hasil pemberitahuan email yang diterima ketika Lambda mendeteksi suhu yang tidak normal. Notifikasi tersebut dikirimkan setiap 10 menit sekali sebagaimana pengaturan yang dilakukan pada CloudWatch EventBridge (Gambar 7).

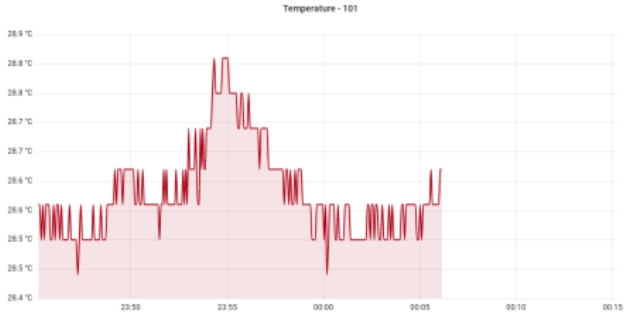
Gambar 7. Email pengujian Lambda dan *Cloud*Watch

C. Hasil Pengujian *Monitoring* Menggunakan Grafana

Monitoring dilakukan untuk memantau nilai suhu dan juga baterai dari perangkat node sehingga dashboard dari website *monitoring* dibuat untuk menampilkan visualisasi dalam bentuk grafik timeseries dan gauge. Grafik timeseries digunakan untuk melihat naik atau turunnya besaran suhu sedangkan gauge digunakan untuk melihat besaran suhu dan tegangan baterai terkini. Hasil konfigurasi dashboard *monitoring* dapat dilihat pada Gambar 8. Untuk grafik timeseries, sumbu x memberikan informasi mengenai waktu sedangkan sumbu y memberikan informasi mengenai besaran suhu yang tercatat.

Gambar 8. Dashboard *monitoring* cold-chain

Selain melakukan *monitoring*, apabila suatu pengiriman cold-chain logistic telah selesai, pengguna dapat melakukan dokumentasi dengan melakukan eksport data ke dalam format CSV. Dalam file CSV tersebut akan berisikan log data suhu atau baterai. Selain itu, agar bentuk terakhir grafik selama pengiriman dapat disimpan atau terdokumtasi dengan baik, pengguna dapat melakukan eksport grafik ke dalam file berformat PDF (Gambar 9).



Gambar 9. Hasil Pengujian Grafana Reporter

V. KESIMPULAN

Dari pengujian yang telah dilakukan serta hasil yang didapatkan maka melalui penelitian ini dapat disimpulkan bahwa:

- Pengintegrasian perangkat gateway dan *cloud* dengan memanfaatkan layanan *serverless* yaitu AWS IoT berhasil dilakukan dan menghasilkan perfoma yang baik dibuktikan dengan hasil pengujian QoS berstandar TIPHON yang berkategorikan memuaskan.
- Sistem arsitektur *serverless* yang dirancang melalui uji fungsionalitas yang dilakukan telah terbukti berhasil menjalankan semua fungsinya yaitu menerima, menyimpan, menampilkan data serta mengirimkan notifikasi ke pengguna melalui email.
- Data yang disimpan pada sistem arsitektur *serverless* yang dirancang dapat di*monitoring* menggunakan Grafana.

DAFTAR PUSTAKA

[1] A. W. Burange and H. D. Misalkar, "Review of Internet of Things in development of smart cities with data management & privacy," 2015 International Conference on Advances in Computer Engineering and Applications, 2015, pp. 189-195, doi: 10.1109/ICACEA.2015.7164693.

[2] Q. Huang and X. Yan, "On the Application of Internet of Things (IOT) in Cold Chain Logistics Management," in Proceedings of the 2018 International Symposium on Communication Engineering & Computer Science (CECS 2018), Hohhot, China, 2018. doi: 10.2991/cecs-18.2018.80.

[3] D. Oktaviani, F. S. Papilaya, and P. F. Tanaem, "Perancangan Aplikasi E-Menu Restaurant dengan Menggunakan Cloud Computing dan Serverless Architecture Lambda," Explore. jurnal. sistem. inf. dan. telematika, vol. 12, no. 1, p. 1, Apr. 2021, doi: 10.36448/jsit.v12i1.1887.

[4] H. A. Rochman, R. Primananda, and H. Nurwasito, "Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol MQTT pada Smarthome," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 1, no. 6, pp. 445–455, 2017.

[5] Aprianto Budiman, M. Ficky Duskarnaen, and Hamidillah Ajie, "ANALISIS QUALITY OF SERVICE (QOS) PADA JARINGAN INTERNET SMK NEGERI 7 JAKARTA," pinter, vol. 4, no. 2, pp. 32–36, Dec. 2020, doi: 10.21009/pinter.4.2.6.