**CESS (Journal of Computer Engineering System and Science)** **p-ISSN :2502-7131**
**Vol. 6 No. 1 Januari 2021** **e-ISSN :2502-714x**

# IMPLEMENTASI *FOG COMPUTING* PADA APLIKASI *SMART HOME* BERBASIS *INTERNET OF THINGS*

**Ahmad Zainudin[1], Ida Anisah[2], Melki Mario Gulo[3]**



*[123]Politeknik Elektronika Negeri Surabaya*
*Jl. Raya ITS – Kampus PENS Sukolilo, Surabaya, Indonesia*
*[1]zai@pens.ac.id, [2]ida@pens.ac.id, [3]melkimariogulo@gmail.com*

*Abstrak*— **Teknologi Internet of Things (IoT) saat ini terus berkembang dan manfaatnya sudah mulai bayak dirasakan oleh sebagian besar masyarkat. Beberapa aplikasi IoT seperti smart home, smart factory, smart agriculture. Pada implementasi sistem IoT diperlukan perangkat yang berfungsi untuk mengumpulkan dan memproses beberapa jenis data. Sehingga diperlukan sebuah resouce yang handal yang sering disebut dengan cloud computing. Cloud computing merupakan pusat data yang terpusat. Karena jarak yang jauh maka menjadi kelemahan untuk beberapa aplikasi yang sensitif terhadap waktu. Pada penelitian ini akan dikembangkan sebuah sistem fog computing pada internet of things services pada untuk aplikasi smart home. Berdasarkan hasil pengujian didapatkan waktu proses komputasi pada aplikasi monitoring suhu dan kelebaban sebesar 0,152 detik, pada aplikasi pengaturan dimmer lampu sebesar 0,339 detik dan apada aplikasi *face recognition* sebesar 6,602 detik.**

*Kata Kunci*— **fog computing, raspberry pi, smart home, iot**

*Abstract*— **Currently, Internet of Things (IoT) is continuing to develop and the benefits can be felt by society. Several IoT applications such as smart home, smart factory, smart agriculture. Cloud computing is a centralized data center which have delay problem. Fog computing can improve the performance for the time-sensitive application. In this research, a fog computing system will be developed on the internet of things services for smart home applications. Based on the test results, the computation process time in fog server for temperature monitoring application is 0.152 seconds, the lamp dimmer control application is 0.339 seconds and in the face recognition application is 6.602 seconds**

*Keywords*— **fog computing, raspberry pi, smart home, iot**

## I. PENDAHULUAN

Teknologi Internet of Things (IoT) saat ini terus berkembang dan manfaatnya sudah mulai bayak dirasakan oleh sebagian besar masyarkat. Beberapa aplikasi IoT seperti *smart home, smart factory, smart agriculture*. IoT ini berawal dari beberapa perangkat pintar seperti telepon pintar, jam tangan pintar hingga beberapa alat yang tidak dapat berkomunikasi seperti lampu, AC dan lain-lain dapat terhubung internet. Beberapa perangkat pada IoT tidak hanya mengirim data tetapi juga menyimpan dan memproses data. Menggunakan beberapa sensor untuk melakukan proses *sensing* kondisi lingkungan dan melakukan aktifitas secara berkala melalui sistem aktuator. Masyarakat terlibat pada ekosistem ini dengan mengakses data melalui telepon pintar atau perangkat lainnya yang dapat digunakan.

Pada implementasi sistem IoT diperlukan perangkat yang berfungsi untuk mengumpulkan dan memproses beberapa jenis data. Selain itu juga bertugas untuk melakukan eksekusi menampilkan secara visual atau melakukan aktuasi. Tugas-tugas ini tidak dapat dilakukan perangkat *node* IoT yang memiliki keterbatasan sumber daya penyimpanan, keterbatasan kemampuan komputasi, keterbatasan *network resources* dan keterbatasan catu daya. Sehingga diperlukan sebuah sumber daya yang handal yang sering disebut dengan komputasi awan. Komputasi awan merupakan pusat data yang terpusat. Karena jarak yang jauh maka menjadi kelemahan untuk beberapa aplikasi yang sensitif terhadap waktu.

Untuk beberapa aplikasi yang sensitif terhadap *latency*/waktu dapat menerapkan *fog computing*. *Fog computing* merupakan paradigma baru dengan melebarkan kemampuan bebasis cloud pada jaringan *edge*. Dengan kata lain mendistribusikan tugas penyimpanan data dan pemrosesan data pada perangkat *fog*. Dengan penerapan *fog computing* ini juga dapat mengurangi biaya untuk penyewahan server cloud.

Beberapa penelitian sudah dilakukan untuk menguji kenerja sistem *fog computing* ini baik melalui simulasi maupun pendekan melalui implementasi secara langsung. Tariq, Q; dkk pada tahun 2018 mengusulkan sebuah fog simulator dengan nama FogNetSim++ yang bersifat terbuka/*open souce*. Melalui software simulator ini peneliti dapat melakukan pengaturan dan penambahan algoritma

managemen *node* seperti algoritma penjadwalan dan mekanisme managemen handover. Algoritma yang ditawarkan dianalisa berdasarkan skalabilitas dan keefektifan penggunaan CPU dan memori. Selain itu juga dianalisa beberapa parameter jaringan diantaranya *delay* eksekusi, *packet error rate*, *handover* dan *latency*. FogNetSim++ mendukung pemodelan jaringan besar yang dilengkapi dengan fitur mobility. [1]

Jose Santos pada tahun 2019 mengimplementasi arsitektur *fog network* dengan menggunakan sebuah platform container open-source Kubernetes dengan membagi menjadi beberapa *micro-services*. Pada penelitian ini dilakukan beberapa analisa diantaranya parameter waktu respon, konsumsi energi, *latency* jaringan, kehandalan, penggunaan badwidth dan mobilitas. Berdasarkan hasil pengujian yang dilakukan dengan menginstal container pada jaringan *fog* dapat menurunkan *latency* sampai 70% [2].

Arsitektur *fog computing* juga diulas oleh Kamil Faqih, dkk. Pada penelitian ini ditekannkan mengenai usulan infrastutur *fog computing* yang terbagi menjadi beberapa lapis diantaranya data akuisisi layer (DAL), event classification layer (ECL), information mining layer (IML), pengambilan keputusan layer (DML) dan cloud storage layer (CSL). Untuk mencapai layanan apliaksi yang efisien maka setiap lapisan melakukan fungsi yang diperlukan.[3]

M. K. Husein [4] membahas mengenai distribusi beban komputasi pada sistem fog computing atau disebut sebagai load balancing. Sistem ini diterapkan pada aplikasi IoT dengan perangkat dalam jumlah banyak untuk melakukan pemrosesan data. Untuk mengatasi permasalah diterapkan algoritma penjadwalan ant colony optimization (ACO) dan particle swarm optimization (PSO). Hasil pengukuran menunjukkan bahwa ACO dapat meningkatkan waktu respon pada sistem IoT dibandingkan PSO dan round robin (RR) algorithm. Sehingga ACO secara efektif sebagai pembagi atau penyeimbang komputasi pada perangkat fog.

X. Zhao[5] mengusulkan sebuah framework baru microservice container fog system (MSCFS) yang perjalan pada perangkat bergerak dan pada aplikasi yang rentan terhadap delay dengan cost yang minimum. System ini juga mengenalkan framework cost aware computational offloading and task scheduling (CACOTS) yang dapat menyelesaikan penjadwalan proses kedalam beberapa step pekerjaan. Berdasarkan hasil pengujian menujukkan bahwa usulan skema MCCFS dan CACOTS dapat meningkatkan kinerja server fog. Hal ini ditunjukkan dengan menurunnya latency layanan, rata-rata waktu bootup layanan lebih efektif dan cost yang minimum.

Pada penelitian ini dikembangkan sebuah arsitektur fog computing yang diterapkan untuk aplikasi smart home. Pada aplikasi smart home ini mengimplementasikan machine learning untuk menjalankan beberapa fitur diantaranya penentuan kondisi kenyamanan ruangan berdasarkan data pengukuran suhu. Dari nilai suhu ini diklasifikasikan kedalam kondisi cukup nyaman, nyaman dan sangat nyaman. Untuk fitur yang lain adalah pengaturan dimmer lampu ruangan berdasarkan kecerahan cahaya ruanan. Selain itu juga pada system smart home ini dilengkapi dengan fitur pendeteksi wajah sebagai keamanan rumah.

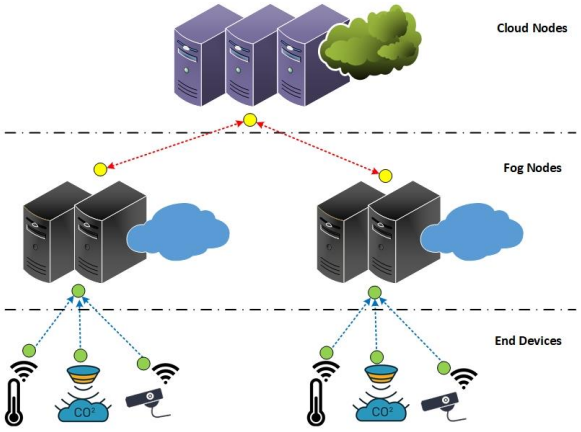
## II. FOG COMPUTING PADA APLIKASI IOT

### A. *Internet of Things*

Internet of Thing (IoT) adalah sebuah konsep dimana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan tanpa memerlukan interaksi manusia ke manusia atau manusia ke komputer. IoT telah berkembang dari konvergensi teknologi nirkabel, micro-electromechanical systems (MEMS), dan Internet. “A Things” pada Internet of Things dapat didefinisikan sebagai subjek misalkan orang dengan monitor implant jantung, hewan peternakan dengan transponder biochip, sebuah mobil yang telah dilengkapi built-in sensor untuk memperingatkan pengemudi ketika tekanan ban rendah. Sejauh ini, IoT paling erat hubungannya dengan komunikasi machine-to-machine (M2M) di bidang manufaktur dan listrik, perminyakkan, dan gas. Produk dibangun dengan kemampuan komunikasi M2M yang sering disebut dengan sistem cerdas atau “smart”. Sebagai contoh yaitu smart kabel, smart meter, smart grid sensor. Penelitian pada aplikasi IoT masih dalam tahap perkembangan. Oleh karena itu, tidak ada definisi dari Internet of Things. Menurut Ashton pada tahun 2009 definisi awal IoT adalah Internet of Things memiliki potensi untuk mengubah dunia seperti pernah dilakukan oleh Internet, bahkan mungkin lebih baik.

### B. *Fog Computing*

Fog computing pertama dikenalkan oleh Cisco. Dibandingkan dengan pusat data could, fog menyediakan sebuah lingkungan komputasi secara virtual yang dijalankan antara cloud dan end users. Cloud dan fog mempunyai fungsi yang sama untuk melayani user, hanya saja fog untuk melayani daerah spesifik tertentu. Fog computing dikembagkan untuk aplikasi IoT yang sensitif terhadap delay. Pada cloud semakin jauh jarak cloud server dengan end user maka latency akan tinggi pula. Gambaran umum arsitektur fog computing seperti terlihat pada gambar 1.




Gbr. 1 Gambaran umum arsitektur fog computing.

Berdasarkan blok diagram diatas fog nodes terletak diantara cloud node dan end devices. Fog nodes dapat juga dikatakan sebagai smart gateway. Pada fog nodes ini data sensing dari end devices tidak hanya dilanjutkan di cloud nodes tetapi juga dilakukan pemrosesan data apabila secara beban komputasi fog node tidak dalam kondisi tinggi. Proses komputasi dilakukan secara terdistribusi pada fog nodes dan cloud nodes. Fog dapat dijalankan pada perangkat jaringan edge seperti router, access point, Road Side Units (RSUs) dan perangkat lainnya. Dengan pengembangan node fog maka kehandalan, toleransi kegagalan dan skalabilotas dapat diatur dengan mudah[8]. Penerapan fog computing ini lebih tepat untuk penerapan Smart Gateway.

### C. *MQTT (Message Queuing Telemetry Transport)*

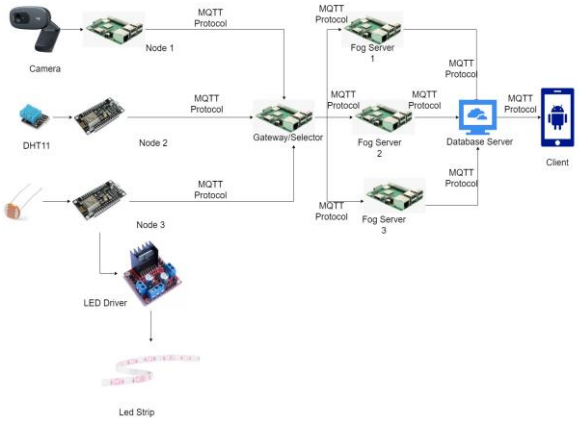
MQTT protokol merupakan sebuah protokol yang berjalan diatas stack TCP/IP dan dirancang khusus untuk machine to machine yang tidak memiliki alamat khusus. Sistem kerja MQTT menerapkan :

- Publish merupakan cara suatu device untuk mengirimkan datanya ke Subscribes. Biasanya pada Publish ini adalah sebuah device yang terhubung dengan sensor tertentu.
- Subscribe merupakan cara suatu device untuk menerima berbagai macam data dari Publish. Subscribe dapat berupa aplikasi monitoring sensor dan sebagainya, Subscribe ini yang nantinya akan meminta data dari Publish.
- Broker pada MQTT berfungsi untuk menghandle data publish dan subscribe dari berbagai device, bisa diibaratkan sebagai server yang memiliki alamat IP khusus. Beberapa contoh dari Broker yang ada seperti Mosquitto, HiveMQ dan Mosca.
- Topic seperti halnya pengelompokan data disuatu kategori tertentu. Pada sistem kerja MQTT protokol ini, topic bersifat wajib hukumnya. Pada setiap transaksi data antara Publish dan Subscribe harus memiliki suatu topic tertentu.[5]

## III. METODE PENELITIAN

### A. *Perancangan Sistem*

Sistem IoT yang dibuat pada penelitian ini adalah untuk mengetahui kehandalan sebuah sistem komunikasi pada sebuah jaringan sensor nirkabel dan IoT yang yang dilengkapi dengan kecerdasan buatan (*artificial intelligence).* Pada sistem ini terdapat 3 perangkat node sensor yang digunakan. Diantaranya adalah node sensor untuk pengukuran suhu dan kelembaban, node sensor untuk pengukuran intensitas cahaya luar dengan aktuator led yang diatur kecerahannya, serta node kamera yang digunakan untuk keamanan pintu masuk rumah dengan melakukan proses pendeteksian wajah. Gambaran umum perancangan sistem seperti terlihat pada gambar 1.




Gbr. 2 Rancangan sistem

Berdasarkan gambar 2 menunjukkan bahwa node 1 merupakan node untuk pendeteksi wajah. Node ini terdiri dari kamera dan perangkat raspberry pi. Kamera digunakan untuk meng-capture wajah dan akan diproses pendeteksian wajah pada raspberry pi. Ini merupakan salah satu fitur aplikasi smart home yang sudah dikembangkan dengan tujuan untuk mendeteksi apabila ada seseorang yang akan masuk rumah bukan anggota keluarga, maka akan dikirim notifikasi pada smatrtphone pengguna aplikasi ini. Pada node 1 apabila terdeketsi *object* wajah yang ter-*capture* oleh kamera dan dilakukan pengambilan gambar. Hasil *capture* gambar akan dikirim ke server fog untuk dilakukan proses deteksi wajah. Sedangkan pada node 2 merupakan node yang berfungsi untuk mendeteksi kondisi suhu ruangan. Node ini dilengkapi dengan sensor DHT11 yang berfungsi untuk mengukur suhu dan kelembaban ruangan. Data suhu dan kelembaban diambil secara berkala dan dikirim ke fog server untuk dilakukan kasifikasi apakah termasuk kondisi sangat nyaman, cukup nyaman dan nyaman.

Pada node 3 digunakan untuk pengatur dimmer lampu ruangan berdasarkan nilai kecerahan cahaya. Untuk mengetahui tingkat kecerahan cahaya, node ini dilengkapi dengan sensor LDR. Data nilai sensor LDR yang terbaca dikirim ke fog server dilakukan proses

klasifikasi. Hasil keputusannya akan digunakan untuk mengatur kecerahan dari lampu LED strip.

Sistem ini memiliki topologi seperti gambar 1, dimana setiap node sensor akan terhubung dengan perangkat gateway. Gateway ini merupakan perangkat raspberry pi 4 yang berfungsi sebagai *selector* untuk memilihkan perangkat fog server terdekat dengan status idle dan siap menjalankan pekerjaan (*task*). Pada sistem ini terdapat 3 pekerjaan yaitu pendeteksi wajah, pendeteksi suhu ruangan dan pengatur kecerahan lampu ruangan. Beberapa proses pekerjaan ini disiapkan dan tertanam pada semua perangkat fog server. Tetapi dalam proses pekerjaannya semua pekerjaan tidak dijalankan pada perangkat fog server yang sama. Masing-masing perangkat fog server menjalankan 1 pekerjaan saja pada waktu yang bersamaan. Pemilihan perangkat fog sever dan jenis pekerjaannya dilakukan secara acak oleh *selector*. Sehingga saat data diterima oleh perangkat *selector* dari node sensor maka selector akan memelih perangkat fog server yang memproses. Node server yang dipilih adalah perangkat fog server yang sedang tidak menjalankan tugas. Fungsi dari pembagian tugas ini ada sebagai load balancing dengan tujuan untuk mendapatkan nilai waktu proses yang lebih kecil. Sehingga pada perangkat selector terdapat list status dari masing-masing fog server apakah dalam kondisi idle (tidak menjankan tugas) atau sedang menjalankan tugas.

Media jaringan yang digunakan untuk menghubungkan antar perangkat adalah WiFi, serta protokol pengiriman data yang digunakan adalah MQTT. Setiap Node sensor akan mengirimkan datanya ke gateway server, setelah data diterima oleh gateway server selanjutnya akan dilakukan *queueing* data dengan metode FIFO untuk selanjutnya dikirimkan ke Fog Server. Sistem ini memiliki 3 Fog Server sebagai pengolah data. Fungsi FIFO pada gateway atau *selector* akan menentukan apakah data dari node sensor akan dikirimkan pada fog server 1, 2 atau 3. Data yang diterima pada fog server akan dimasukan kedalam sebuah model machine learning untuk menghasilkan sebuah output tertentu. Terdapat 3 model machine learning yang digunakan sebagai fungsi deterministik terhadap sebuah keputusan tertentu. Ketiga model machine learning tersebut sudah tertanam pada masing-masing perangkat fog server. Gambaran proses ketiga model machine learning secara singkat seperti penjelasan di bawah ini.

1. Model penentuan kondisi kenyamanan sebuah ruangan dengan nilai masukan berupa data suhu dan kelembaban sebuah ruangan. Proses model machine learning ini menghasilkan luaran atau keputusan berupa 3 state kondisi (cukup nyaman, nyaman, sangat nyaman).
2. Model penentuan nilai dimmer lampu dengan data masukkan berupda nilai kecerahan cahaya yang terukur menggunakan sensor LDR pada sudah terpasang pada node sensor, dan Output berupa nilai analog PWM yang akan digunakan pada LED strip sebagai pengatur dimmer lampu.
3. Model face recognition yang digunakan sebagai sistem keamanan rumah, dimana data masukan yang digunakan adalah foto yang diambil secara otomatis oleh node kamera dan luarannya berupa binary classification (benar atau salah)

Setelah Data masuk pada fog server dan dilakukan deterministik hasil sesuai dengan model yang digunakan. selanjutnya data tersebut akan dikirimkan ke database server. Database server digunakan sebagai endpoint server yang berhubungan dengan klien. Pada client berjalan aplikasi mobile android yang berfunsi sebagai user interface untuk menampilkan data monitoring meliputi status nilai suhu ruangan, kecerahan cahaya pada ruangan dan menampilkan foto wajah yang ter-capture oleh kamera. Hasil pengenalan wajah ini akan menampilkan status wajah merupakan anggota keluarga atau bukan. Aplikasi mobile pada client dapat diakses dimana saja. Prinsip kerja dari aplikasi mobile ini adalah melakukan subscribe topic mqtt server tertentu ke cloud server. Pada cloud server ini berjalan database server.

### *B. Implementasi Sistem*

Pada implementasi sistem terbagi menjadi implementasi perangkat keras dana perangkat lunak. Pada implementasi perangkat keras dilakukan konfigurasi pada sisi perangkat node sensor, node selector atau gateway, fog server, database server dan aplikasi mobile pada user. Pada bagian implementasi perangkat lunak lebih ditekankan pada implementasi artificial intelligent yang tertanam pada fog server. Training ketiga model dilakukan pada fog server menggunakan beberapa metode yang berbeda.

Pada proses training model penentuan kondisi kenyamanan sebuah ruangan, langkah pertama dilakukan pengumpulan data suhu dan kelembaban di sebuah ruangan tertentu. Pengumpulan data dilakukan dengan cara mengirimkan nilai suhu dan kelembaban yang terbaca pada node sensor ke sebuah server database. Dari data yang telah diambil selama 2 hari, selanjutnya dilakukan pengolahan dengan menggunakan pemrograman python dan library scikit-learn. Scikit-learn merupakan library yang digunakan untuk melakukan pembuatan model machine learning. Pada pembuatan model dengan menggunakan scikit-learn dan python tersebut, digunakan metode clustering dengan K-Means. Metode clustering digunakan karena dari data yang didapatkan dari node sensor tidak memiliki label dan tidak dilakukan pelabelan. Sehingga dengan metode K-Means, data akan di cluster berdasarkan nilai-nilai terdekatnya. Jumlah cluster yang digunakan sebanyak 3 buah, untuk merepresentasikan 3 buah state yaitu cukup nyaman, nyaman, sangat nyaman. Setelah didapatkan Cluster, dilakukan pelabelan pada cluster sehingga

data yang didapatkan bisa merepresentasikan state yang diinginkan.

Pada proses training model penentuan nilai dimmer lampu, pengumpulan data dilakukan dengan cara mengukur nilai kecerahan cahaya luar dengan menggunakan LDR, dan melakukan kontroling kecerahan LED strip dengan menggunakan potensiometer. Nilai LDR dan potensiometer yang digunakan untuk mengontrol LDR dikirimkan ke server database. Pengambilan data dilakukan selama 2 hari. Data yang telah ter record pada server database selanjutnya dilakukan pengolahan dengan python dan scikit-learn menggunakan metode random forest dengan jumlah estimator sebanyak 20. Random forest digunakan karena data yang diolah adalah nilai regresi. Selanjutnya estimator random forest disimpan menjadi sebuah model yang digunakan pada sistem ini.

Untuk proses training pada model face recognition dilakukan pengumpulan data berupa beberapa foto atau gambar wajah dari orang. Disini foto orang tersebut diolah menggunakan python, scikit-learn, tensorflow, serta Opencv. Opencv merupakan sebuah library pada python yang digunakan untuk melakukan pemrosesan citra. Sedangkan tensorflow merupakan sebuah deep learning framework pada python. Pada proses pengolah data, pertama dilakukan croping foto wajah seseorang tersebut dengan menggunakan algoritma Haar Cascade face Classifier oleh Program. Selanjutnya foto yang telah di lakukan croping, akan masuk kedalam fungsi image generator tensor flow guna mendapatkan variasi input yang bermacam-macam dari foto. Variasi ini dapat berupa efek bluring, size scaling, rotating, serta color scaling pada foto. Selanjutnya dilakukan training model dengan foto yang telah di generate tersebut. Training model menggunakan arsitetktur Facerec.net . Facerec.net adalah sebuah arsitektur deep learning yang digunakan untuk melakukan pembuatan model face recognition. Setelah proses training selesai selanjutnya dilakukan penyimpanan model training sehingga dapat digunakan pada fog server.

## IV. HASIL DAN ANALISA

Pengujian dilakukan untuk mengetahui dari kinerja sistem yang sudah dikembangkan. Pada penelitian ini dilakukan pengujian waktu delay proses masing-masing aplikasi (monitoring suhu dan kelembaban, pengaturan dimmer lampu LED dan *face recognation*). Pengujian dilakukan dengan beberapa scenario diantaranya pengukuran delay proses dengan menggunakan broker mqtt jaringan internet dan broker pada jaringan lokal.

### *A. Pengukuran delay proses pada fog server dengan menggunakan broker mqtt jaringan internet*

Pada pengujian ini bertujuan untuk mengetahui delay proses pada fog server pada semua aplikasi. Nilai delay proses ini didapatkan pada masing-masing fog server untuk masing-masing aplikasi dan dikirim ke database server. Data hasil pengujian delay untuk scenario menggunakan broker mqtt jaringan internet seperti terlihat pada table 1.

TABEL I
PENGUKURAN DELAY PADA FOG SERVER UNTUK SKENARIO BROKER MQTT JARINGAN INTERNET

| Pengujian ke- | Monitoring suhu dan selembaban (s) | Pengaturan dimmer lampu LED (s) | *Face recognation* (s) |
|---|---|---|---|
| 1 | 0,162 | 0,464 | 7,610 |
| 2 | 0,157 | 0,447 | 7,720 |
| 3 | 0,166 | 0,557 | 7,750 |
| 4 | 0,156 | 0,467 | 7,760 |
| 5 | 0,161 | 0,598 | 7,020 |
| 6 | 0,163 | 0,564 | 7,802 |
| 7 | 0,159 | 0,551 | 7,183 |
| 8 | 0,154 | 0,568 | 7,810 |
| 9 | 0,151 | 0,449 | 7,840 |
| 10 | 0,163 | 0,430 | 7,740 |
| Rata-rata | 0,159 | 0,510 | 7,624 |

Berdasarkan data hasil pengukuran pada table 1 menunjukkan rata-rata proses komputasi pada fog server untuk aplikasi monitoring suhu dan kelembaban sekitar 0,159 detik. Data ini merupakan hasil kalisifikasi dengan menggunakan K-Means yang berjalan pada perangkat raspberry pi 4. Untuk pengaturan dimmer lampu LED didapat hasil pegukuran waktu komputasinya sekitar 0,510 detik. Data yang diproses berupada data dari sensor LDR. Pada aplikasi pengaturan dimmer lampu LED ini digunakan metode *random forest* dengan jumlah estimator sebanyak 20 dengan data yang diolah nilai regresi. Sedangkan pada aplikasi face recognition didapatkan nilai waktu komputasinya sekitar 7,624 detik. Aplikasi *face recognice* ini menggunakan tensorflow dan Opencv.

### *B. Pengukuran delay proses pada fog server dengan mengguankan broker mqtt jaringan lokal*

Pengujian di bawah ini bertujuan untuk mengetahui besarnya delay proses pada fog server dengan menggunakan broker mqtt pada jaringan lokal. Hal ini dilakukan dengan menginstall aplikasi broker mosquito pada perangkat raspberry pi 4. Sehingga setiap data dari node sensor yang dikirim ke fog server dan hasil keputusan dari fog server diakses oleh aplikasi mobile melalui mqtt broker lokal.

TABEL II
PENGUKURAN DELAY PADA FOG SERVER UNTUK SKENARIO BROKER MQTT JARINGAN INTERNET

| Pengujian ke- | Monitoring suhu dan selembaban | Pengaturan dimmer lampu LED | *Face recognation* |
|---|---|---|---|
| 1 | 0,152 | 0,302 | 6,512 |
| 2 | 0,143 | 0,313 | 6,562 |
| 3 | 0,145 | 0,315 | 6,653 |
| 4 | 0,170 | 0,323 | 6,617 |
| 5 | 0,140 | 0,328 | 6,646 |

| | | | |
|---|---|---|---|
| 6 | 0,160 | 0,374 | 6,583 |
| 7 | 0,147 | 0,341 | 6,588 |
| 8 | 0,148 | 0,379 | 6,544 |
| 9 | 0,163 | 0,354 | 6,665 |
| 10 | 0,152 | 0,361 | 6,648 |
| Rata-rata | 0,152 | 0,339 | 6,602 |



Berdasarkan hasil pengukuran pada tabel 2 menujukkan bahwa waktu proses aplikasi monitoring pada fog server dengan metode K-Means diperlukan waktu sekitar 0,152 detik untuk memprosesnya. Untuk aplikasi pengaturan dimmer lampu LED diperlukan waktu proses sebesar 0,339 detik dengan metode *random forest*. Sedangkan waktu proses aplikasi *face recognition* dengan menggunakan tensorflow dan Opencv membutuhkan waktu proses sekitar 6,602 detik.

## V. KESIMPULAN

Bersadarkan hasil pengujian yang sudah dilakukan dapat disimpulkan bahwa pengunaan mqtt broker pada jaringan lokal bisa menggurangi waktu komputasi. Pada skenario broker mqtt pada jaringa lokal didapatkan waktu proses aplikasi monitoring suhu dan kelebaban sebesar 0,152 detik, pada aplikasi pengaturan dimmer lampu sebesar 0,339 detik dan apada aplikasi face recognition sebesar 6,602 detik. Besar waktu komputasi yang didapatkan masih relative cukup besar. Hal ini bisa disebabkan keterbatasan spesifikasi dari perangkat raspberry pi 4 untuk melakukan proses komputasi artificial intelligent. Untuk meningkatkan performasi fog server dapat digunakan perangkat fog server dengan spesifikasi yang lebih tinggi.

## UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Pihak Manajemen Kampus PENS yang telah memberi dukungan dana terhadap pelaksanaan penelitian ini melalai Skema Dana Penelitian Lokal Tingkat Dasar Tahun 2020 dengan nomer SK : 1778/PL14/PG/2020.

## REFERENSI

[1] T. Qayyum, A. W. Malik, M.A K. Khattak, O. Khalid, S.U Khan, FigNetSim++: A Toolkit for Modeling and Simulation of Distributed Fog Environment, IEEE Access, 2018

[2] J. Santos, T. Wauters, B. Volckaert, F. D. Turck, Resource in Fog Computing : From Teory to Practice, Sensor MDPI, 2019

[3] K. Faqih, M. Faris Labib, A, N. Afandi, Arsitektur Fog Computing Smart Home Berbasis Internet of Things (IoT), Seminar Nasional Fortei Regional 7, Universitas Negeri Malang, 2019

[4] M. K. Hussein, M. H. Mousa, Efficient Task Offloading for IoT-Based Application in Fog Computing Using Ant Colony Optimation, IEEE Access, 2020

[5] X. Zhao, C. Huang, Microservice Based Computation Offloading Framework and Cost Efficient Task Scedulling Algorithm in Heterogenous Fog Cloud Network.