Implementasi Monitoring Smart Building Berbasis Website Pada Prototipe Sistem Otomasi dan Pemantauan Smart Building

1st Raihan Fadilla Hakim
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
raihanfhkm@student.telkomuniversity.ac.id

2nd Favian Dewanta
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
favian@telkomuniversity.ac.id

3rd Bagus Aditya
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
goesaditya@telkomuniversity.ac.id

***Abstrak* — *Smart building* merupakan salah satu teknologi yang sedang berkembang pesat pada masa ini. *Smart building* merupakan salah satu contoh implementasi dari IoT (*Internet of Things*), di mana konsepnya adalah mengintegrasikan teknologi informasi dan komunikasi dengan objek-objek fisik seperti bangunan, kendaraan dan masih banyak hal lain lagi, dan semua itu terhubung dengan akses internet. Salah satu masalah besar saat membangun *smart building* adalah keamanan serta penggunaan energi yang tidak terkontrol sehingga menimbulkan pemborosan energi yang sia-sia. Untuk itu penelitian ini melakukan pembuatan prototipe sistem *monitoring* dan otomasi pada *smart building* dengan menggunakan MQTT (*Message Queuing Telemetry Transport*) protokol komunikasi yang di enkripsi sebagai keamanan data antara mikrokontroler yang terhubung dengan Raspyberry Pi sebagai server terhubung dengan sensor yang selalu mengirimkan data secara *real-time* dan *website monitoring* akan menampilkan data suhu dan kelembaban. Pada sistem otomasi menggunakan Raspyberry Pi lainnya sebagai client yang akan terhubung dengan Raspyberry Pi sebagai *server* dengan dihubungkan melalui BACNet menggunakan IP *address* masing-masing perangkat lalu relay akan terhubung dengan *client* dan akan menyalakan lampu sebagai pengganti AC (Air Conditioning).**

***Kata kunci :* IoT, Smart Building, MQTT, Raspyberry Pi. PLC, BACNet**

I. PENDAHULUAN

Perkembangan teknologi informasi dan peningkatan populasi manusia beberapa tahun ini berkembang dengan cepat sehingga perkembangan ini merupakan kemajuan teknologi umat manusia seluruh informasi dapat tersedia secara mudah melalui berbagai *platform* sehingga informasi yang tidak terbatas [1]. Perkembangan teknologi salah satunya ialah *smart building* yang merupakan suatu inovasi terbarukan dan dapat digunakan serta diaplikasikan untuk berbagai macam tipe bangunan. *Smart building* telah dikembangkan sejak 1980-an hingga saat ini terus berkembang dengan cepat mengikuti perkembangan teknologi dan inovasi serta perangkat yang tersedia dalam industri elektronik [2]. Penelitian yang ada 40 % penggunaan energi terbesar di Amerika Serikat adalah bangunan dan sebagian besar penghuni bangunan tersebut tidak puas dengan tempat huni mereka. Dengan inovasi *smart building* maka dapat menjadi lingkungan bangunan yang terjangkau dan berkelanjutan dalam skala global dengan perkotaan yang maju [3].

Oleh karena itu, pentingnya teknologi seperti IoT (*Internet of Things*) dibutuhkan dalam pembuatan bangunan menjadi *smart building* untuk menghindari pemborosan energi yang secara berlebihan, membangun strategi manajemen untuk *monitoring* yang efektif, dan pemantauan energi untuk bangunan sangat dibutuhkan untuk mengetahui nilai seperti suhu atau kelembaban jika mengaktifkan AC (*Air Conditioner*) [4].

Tujuan penelitian ini adalah untuk membuat sistem pemantauan berbasis situs *website* HTTP (*Hypertext Transfer – Transfer Protocol*) yang akan memungkinkan pemantauan jarak jauh dan menyederhanakan penggunaan situs web bagi pengguna. Suhu dan kelembaban akan ditampilkan secara *real-time* di *website monitoring* atau *dashboard online* berupa grafik *chart* yang menunjukkan hasil data sensor dan waktu yang dikumpulkan. Untuk membuat *website monitoring* menggunakan bahasa pemrograman Python dengan *framework* Flask yang didukung oleh bahasa pemrograman JavaScript untuk membuat grafik *chart* serta menggunakan MySQL pada XAMPP *Controller* untuk *database* suhu dan kelembaban. Untuk tampilan *website* menggunakan *template* CSS *Bootstrap* yang dimodifikasi sesuai kebutuhan dengan menggunakan *framework* Flask.

II. KAJIAN TEORI

A. Python

Semakin banyak alternatif yang dapat digunakan dalam bahasa pemrograman yang digunakan dalam pengembangan aplikasi. Python merupakan bahasa pemrograman multi tujuan untuk berbagai keperluan yang paling banyak digunakan oleh seluruh pengembang aplikasi, *data science* dan lainnya. Python dapat dipelajari secara mudah dan digunakan dengan memiliki sintaks yang sederhana serta mendukung berbagai *framework* yang ada seperti flask, django, pyramid dan cherrypy [11].

B. *Framework* Flask

Framework Flask merupakan *micro-framework* yang digunakan untuk membuat aplikasi *website* dengan menggunakan bahasa pemrograman Python secara mudah untuk fungsionalitas, fleksibilitas, ringan, dan untuk digunakan [13]. Flask dapat diimplementasikan ke WSGI (*Web Server Gateway Interface*) yang menyediakan *local server* seperti localhost:8000 untuk mensimulasikan *website*

secara HTTP. *Framework* ini dapat digunakan untuk *backend* maupun *frontend* untuk keperluan pembuatan aplikasi *website*. Flask dapat menggunakan plugin yang memberikan kesan bahwa fungsi dan komponen tersebut diimplementasikan oleh Flask itu sendiri. Meskipun Flask disebut *micro-framework*, bukan berarti ia tidak memiliki fungsionalitas. *Microframework* disini maksudnya adalah Flask bertujuan untuk membuat inti aplikasi sesederhana mungkin, namun tetap mudah untuk ditambahkan [6].

C. *Bootstrap*

Bootstrap merupakan CSS (*Cascading Style Sheets*) *framework yang dikembangkan oleh Twitter dan menjadi open source*. CSS *Bootstrap merupakan* front-end *framework* yang menyediakan berbagai banyak fitur dan tampilan secara gratis serta menghasilkan tampilan *website* yang secara otomatis menyesuaikan dengan lebar browser yang digunakan. *Framework* ini sangat populer dalam mengembangkan *website* yang *easy to use*, *mobile-friendly*, dan mudah dikembangkan bagi pengembang aplikasi *website* [11].

D. JavaScript

JavaScript merupakan bahasa pemrograman untuk membuat tampilan website dengan berbagai macam model serta membuat grafik yang dikenal dengan Chart.JS. Bahasa pemrograman tersebut penggunaan dan ketersedian yang luas dan didukung oleh banyak *browser* memungkinkan JavaScript pilihan yang tepat untuk membuat *website monitoring* dengan grafik [12].

E. MySQL

MySQL merupakan *database management system* yang *open source* dengan menggunakan program *Query* yang dimiliki SQL (*Structured Query Language*) serta aplikasi *multiuser* yang sangat kompatibel dan dapat mengirim data melalui tool XAMPP *Controller* sebagai penghubung untuk menjalankan *database* MySQL. Pada MySQL *database* dapat dijalankan dengan localhost/phpmyadmin/ pada *browser* dan dapat dihidupkan secara otomatis tanpa menyalakan atau mematikan dengan menjalankan program yang diinginkan [13].

F. *Apache* Jmeter

Aplikasi perangkat lunak *open source* dengan konfigurasi Java yang banyak digunakan untuk mengukur kinerja dari pada aplikasi *website*. *Apache* Jmeter digunakan mengukur struktur uji *point to point* secara otomatis yang dapat diberikan perintah sesuai dengan keinginan pengujian. Dengan menggunakan *Apache* Jmeter dapat juga mengukur kekuatan dari pada *website* seperti jumlah *user* yang masuk secara bersaman ke *website*, mengukur nilai *throughput*, *delay*, dan *packet loss* [10].

F. *Page Speed Insight*

Merupakan *tools* yang digunakan untuk melakukan pengujian performa pada *website*. Pengujian menggunakan *tools* tersebut agar dapat mengetahui performa halaman untuk perangkat seluler dan desktop. Pengujian dengan *Page Speed Insight* dapat memberikan saran agar performa dari *website* tersebut dapat diperbaiki. Nilai yang dihasilkan pada *Page Speed Insight* yaitu :

1. *First Contentful Pain* bagian penilaian untuk melihat performa waktu saat teks atau tampilan *website* ditampilkan.
2. *Largest Contentful Pain* bagian yang menunjukan hasil waktu saat teks atau gambar terbesar.
3. *Cumulatavie Layout Shift* pengukuran perpindahan elemen pada *website*.
4. *Speed Index* merupakan penilaian yang menunjukan seberapa cepat halaman *website* ditampilkan
5. *Total Blocking Time* waktu interaktif *website* yang sesungguhnya.

G. QoS (*Quality of Service*)

Mekanisme parameter jaringan yang disebut *Quality of Service* (QoS) merupakan suatu layanan yang dapat berfungsi sebagai standar penilaian kualitas untuk melihat rentang perbedaan serta agar menjadi bahan evaluasi. Parameter yang digunakan berdasarkan ITU (*International Telecommunication Union*) yang merupakan standarisasi internasional di bidang telekomunikasi. Parameter yang diukur ialah *throughput, packet loss,* dan *delay*. Standarisasi yang dipakai ialah versi ITU-T G1010 dapat dilihat pada Tabel 1, Tabel 2, dan Tabel 3. Berikut kategorinya:

1. *Throughput*

Throughput adalah kecepatan transfer data. *Throughput* adalah jumlah dari semua kedatangan paket sukses yang diamati di tujuan selama periode waktu tertentu dibagi dengan lamanya periode waktu tersebut. Berikut ini adalah standar kategori *throughput* berdasarkan ITU-T G1010:

TABEL 1.
Kategori *Throughput*

Kategori	Nilai (bps)	Indeks
Sangat baik	>10 kbps	4
Baik	6 kbps – 10 kbps	3
Cukup	1 kbps - 5 kbps	2
Kurang baik	0 kbps -1 kbps	1
Buruk	0 kbps	0

2. *Packet Loss*

Packet loss adalah banyaknya paket yang gagal mencapai tempat tujuan paket tersebut dikirim [5]. Adapun kategori *packet loss* menurut ITU (*International Telecommunication Union)* adalah sebagai berikut:

TABEL 2.
Kategori *Packet Loss*

Kategori	Nilai (%)	Indeks
Sangat Bagus	<3	4
Bagus	4-15	3
Sedang	16-25	2
Buruk	> 25	1

3. *Delay*

Delay adalah waktu yang dibutuhkan sebuah data untuk menempuh jarak dari asal ke tujuan. *Delay* dapat dipengaruhi oleh jarak media fisik, kongesti atau waktu lama proses yang lama. Adapun kategori *delay* ITU (*International Telecommunication Union*) adalah sebagai berikut.:

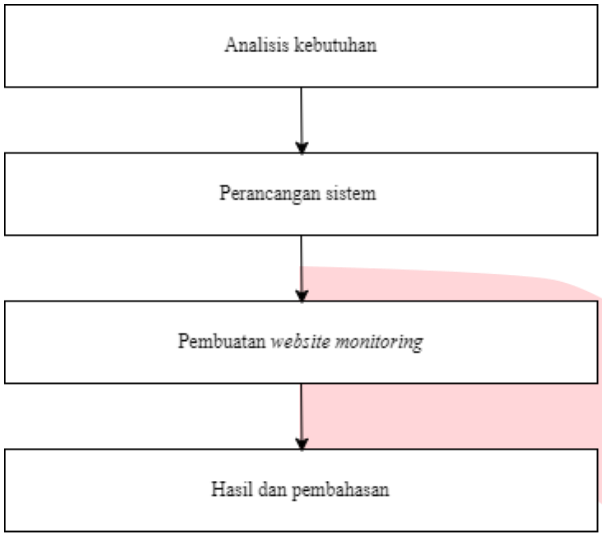
TABEL.3
Kategori *Delay*

Kategori	Nilai (ms)	Indeks
Sangat Bagus	<150	4
Bagus	150-450	3
Sedang	450-650	2
Buruk	>650	1

III. METODE

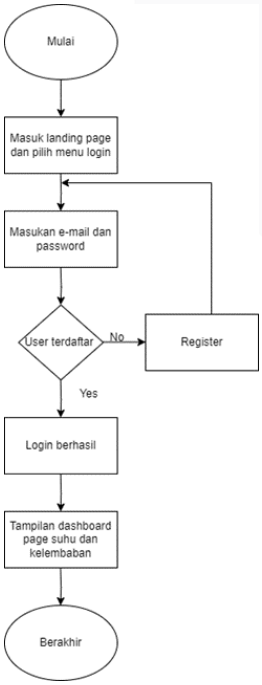
Penelitian ini dilakukan melalui berbagai tahapan untuk membuat *website dashboard monitoring* suhu dan kelembaban agar sesuai dengan kebutuhan.

A. Rancangan Penelitian



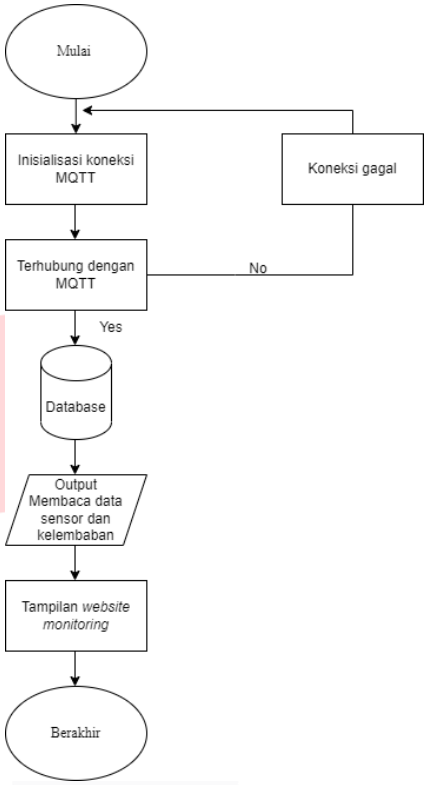
GAMBAR 1.
Rancangan Penelitian

Gambar 1 merupakan rancangan tahapan penelitian yang digunakan untuk analisis kebutuhan. Langkah ini digunakan untuk mengidentifikasi kebutuhan apa saja yang digunakan untuk membuat *website dashboard monitoring* seperti bahasa pemrograman, *framework* yang akan digunakan, *database*, dan hal-lainnya. Tahap kedua dan ketiga merupakan perancangan sistem dan pembuatan *website monitoring*. Tahap ini digunakan untuk merancang sistem untuk membangun dan mengembangkan serta tahap terakhir merupakan hasil dan pembahasan dari tahap-tahap sebelumnya.



GAMBAR 2.
Flowchart Login

Gambar 2 menjelaskan *flowchart* tentang bagaimana proses *login page* berjalan.. *User* akan dapat *login* apabila sudah terdaftar pada *data base* system, jika *user* belum terdaftar maka *user* dapat registrasi secara mudah dengan *input e-mail* dan *password*.



GAMBAR 3.
Diagram alir *website*

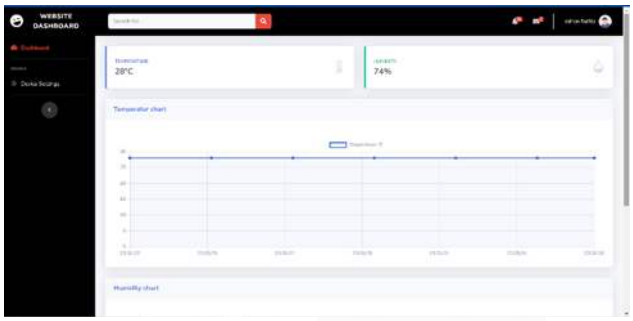
Gambar 3 menunjukan gambaran terkait proses *website monitoring*. *Website* tersebut merupakan *output* data sensor suhu dan kelembaban yang didapat dari MQTT (*Message Queueing Telemetry Transport*) *publisher*. *Website monitoring* ini berfungsi sebagai *subscriber* dari MQTT yang fungsinya menerima data dari *publisher* agar selalu mendapatkan data sensor suhu dan kelembaban apabila sensor dinyalakan. Untuk konfigurasi MQTT menggunakan *broker* "test.mosquitto.org", *port* 1883 dan topik "/Temperature/TA" dan "/Humidity/TA".

IV. HASIL DAN PEMBAHASAN

Tahap awal implementasi dari pembuatan *website monitoring* adalah merencanakan terlebih dahulu kebutuhan jenis data apa saja yang akan ditampilkan, gambaran visualisasi *website* yang akan dibuat dan penampilan grafik diagram dan angka. *Website* dibuat dengan *framework* Flask dan membuat tampilan *landing page*, *login page*, dan *dashboard page* menggunakan *CSS Bootstrap* yang dimodifikasi sesuai kebutuhan. Tampilan tersebut dapat dilihat pada Gambar 4, Gambar 5, dan Gambar 6.

GAMBAR 4.
Home page

GAMBAR 5.
Login page



GAMBAR 6.
Dashboard page

Tahap berikutnya jika tampilan halaman *frontend* telah dirancang dan diimplementasikan selanjutnya membuat *backend* untuk *website* tersebut. *Backend* dirancang sesuai dengan kebutuhan masing-masing tampilan di mana *backend* akan terintegrasi dengan Raspberry Pi yang telah di enkripsi serta akan menjadi *subscriber* dari MQTT (*Message Queuing Telemetry Transport*).

Untuk mengetahui kinerja serta fungsi yang telah diberikan dari *website* tersebut pengujian dilakukan dengan mengikuti tiga tahapan pengujian. Tahapan pertama yaitu menguji fungsi dari *website* tersebut untuk mengetahui keseluruhan fitur berjalan dengan baik, seperti: *login page* meliputi *e-mail address user* dan *password user*, *dashboard page* meliputi grafik suhu dan kelembaban. Tahapan kedua yaitu QoS (*Quality of Service*) yang dilakukan untuk mengetahui performa *website* monitoring meliputi kualitas jaringan yang diukur adalah *throughput*, *packet loss*, dan *delay* menggunakan perangkat lunak *Apache* JMeter.

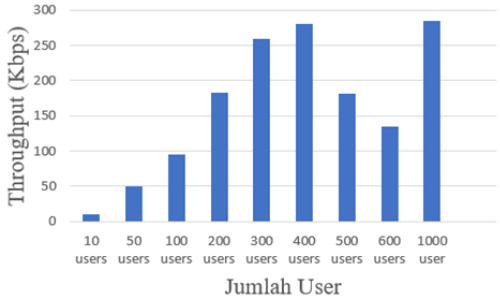
Bagian langkah pengujian ada beberapa tahapan pertama yang dilakukan, yaitu :

1. Jalankan kodingan melalui *visual studio*, dan jalankan localhost:8000 di *web browser*.
2. Setelah terkoneksi dengan localhost, maka tampilan *home page* akan muncul dan pilih menu login.
3. Masukan email address serta password yang terdaftar
4. Dashboard page akan menampilkan *live grafik chart*.

Setelah pengujian dengan mencoba fitur secara langsung maka tahapan berikutnya ialah pengujian dengan menggunakan perangkat lunak *Apache* JMeter. Pengujian QoS (*Quality of Service*) untuk mengetahui performa dari *website monitoring* yang telah dibuat. Pengujian dilakukan dengan cara perhitungan secara jumlah *user* yang masuk kedalam *website* tersebut secara bersamaan. Selanjutnya melakukan pengukuran masing-masing parameter sebagai berikut:

A. *Throughput*

Gambar 7 menunjukan pengukuran menggunakan *Apache* Jmeter untuk mendapatkan hasil *throughput* yang berbeda-beda. Hasil *throughput* tersebut menandakan bahwa *website* berada pada kondisi terbaik untuk menerima jumlah *users* yang masuk ke dalam *website*. Akan tetapi data yang diterima di *website* tidak berpengaruh dengan data yang didapat.. Tabel 4 mendapatkan indeks yang sangat bagus sesuai dengan standar ITU-T G.1010.

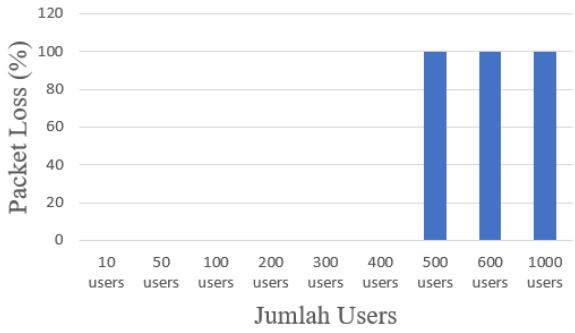


GAMBAR 7.
Hasil *Throughput*

TABEL 4.
Indeks *Throughput*

Jumlah *Users*	*Throughput* (kbps)	Indeks
10 *Users*	10.7	Sangat Bagus
50 *Users*	49.4	Sangat Bagus
100 *Users*	94.8	Sangat Bagus
200 *Users*	182.8	Sangat Bagus
300 *Users*	260.0	Sangat Bagus
400 *Users*	281.1	Sangat Bagus
500 *Users*	181.6	Sangat Bagus
600 *Users*	135.2	Sangat Bagus
1000 *Users*	286.1	Sangat Bagus

B. *Packet Loss*



GAMBAR 8.
Hasil *packet loss*

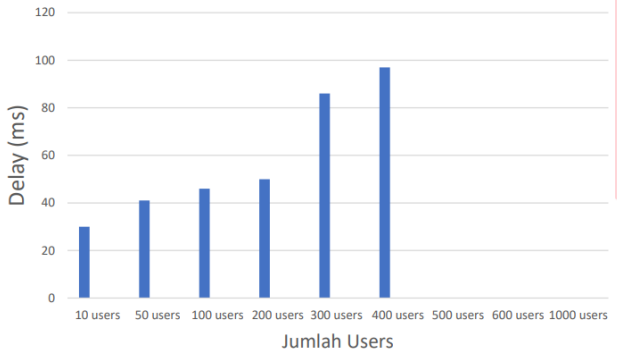
Identifikasi *packet loss* yang didapat setelah menguji performa *website* menggunakan *Apache* Jmeter dapat dilihat pada Gambar 8 hasil yang didapat pada rentang 10 *users* sampai dengan 400 *users* mendapatkan hasil yang sangat bagus yaitu 0 %. Akan tetapi rentang 500 *users* sampai 1000

users hasil yang didapatkan sangat tidak bagus yaitu 100 % dikarenakan pengaruh kapasitas *server* atau jaringan yang tidak stabil dapat mempengaruhi hasil tersebut. Dengan mengikuti standar ITU-T G.1010 dapat dilihat pada Tabel 5.

TABEL 5.
Hasil *packet loss*

Jumlah *Users*	*Packet loss* (%)	Indeks
10 *Users*	0 %	Sangat Bagus
50 *Users*	0 %	Sangat Bagus
100 *Users*	0 %	Sangat Bagus
200 *Users*	0 %	Sangat Bagus
300 *Users*	0 %	Sangat Bagus
400 *Users*	0 %	Sangat Bagus
500 *Users*	100 %	Buruk
600 *Users*	100 %	Buruk
1000 *Users*	100 %	Buruk

C. *Delay*



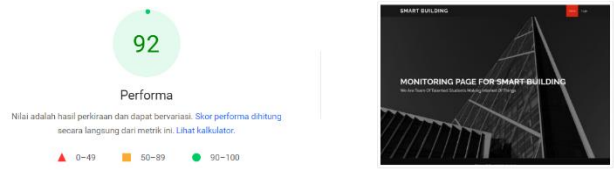
GAMBAR 9.
Hasil *delay* pengukuran

Pengujian *delay* dengan menggunakan *Apache* Jmeter dapat dilihat pada Gambar 9 mendapatkan *delay* dibawah <150 ms. Akan tetapi pada kategori 500 *users* sampai dengan *users* 1000 mendapatkan *delay* 0 ms dikarenakan ketika pengujian berlangsung *website* tidak dapat menerima *request user* dikarenakan terjadinya *traffic network*. *Traffic network* yang terjadi dikarenakan kapasitas server yang tidak bisa menampung begitu banyaknya *request* sehingga terjadinya *down* pada *website* sehingga aliran data yang dikendalikan oleh *bandwith* pada jaringan tidak dapat dikendalikan sehingga terjadinya *network bottleneck.* Jika dilihat pada Tabel 5 indeks yang didapatkan sangat bagus pada rentang 10 *users* hingga 400 *users* dengan catatan pada rentang 500 *users* hingga 1000 *users* mendapatkan indeks yang buruk karena terjadinya *network bottleneck*.

TABEL 6.
Indeks hasil *delay*

Jumlah *Users*	*Delay* (ms)	Indeks
10 *Users*	41.4 ms	Sangat Bagus
50 *Users*	38.56 ms	Sangat Bagus
100 *Users*	64.10 ms	Sangat Bagus
200 *Users*	47.735 ms	Sangat Bagus
300 *Users*	71.54 ms	Sangat Bagus
400 *Users*	83.78 ms	Sangat Bagus
500 *Users*	0	Buruk (terkecuali)
600 *Users*	0	Buruk (terkecuali)
1000 *Users*	0	Buruk (terkecuali)

Apabila pengukuran *Quality of Service* telah dihitung dan dilakukan maka tahap akhir dari pengujian performa *website monitoring smart building* penelitian ini menggunakan *Page Speed Insight*. Performa *website* dapat dilihat pada Gambar 7 yang mendapatkan nilai performa 92 dan kinerja pada *website* yang diakses melalui *desktop* dapat dilihat pada Tabel 7.

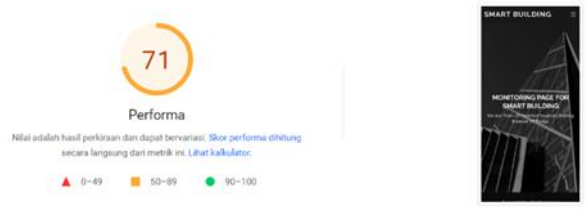


GAMBAR 10.
Pengujian *website* (*desktop*)

TABEL 7.
Indeks Yang Didapat Penilaian

No	Faktor	Nilai	Kategori
1	*First Contentful Pain*	0.8 detik	Sangat Baik
2	*Largest Contentful Pain*	1.1 detik	Sangat Baik
3	*Cumulutavie Layout Shift*	0.052	Sangat Baik
4	*Speed Index*	2.2 detik	Sangat Baik
5	*Total Blocking Time*	0 md	Sangat Baik

Selain penilaian *website* yang diakses melalui *desktop* dapat diakses melalui telepon selular. Penilaian *Page Speed Insight* dapat menilai performa *website* di telepon selular dapat dilihat pada Gambar 11. yang mendapatkan nilai performa 71. Nilai tersebut didapatkan dikarenakan tampilan halaman *website* yang diakses melalui telepon selular tidak dapat menampilkan gambar secara maksimal.



GAMBAR 11.
Pengujian melalui *mobile*

Hasil indeks yang dihasilkan dapat dilihat pada Tabel 8 yang menampilkan 5 faktor serta nilai yang didapatkan.

TABEL 8.
Indeks Yang Didapat Penilaian Melalui *Mobile*

No	Faktor	Nilai	Kategori
1	*First Contentful Pain*	3.2 detik	Baik
2	*Largest Contentful Pain*	5.6 detik	Baik
3	*Cumulutavie Layout Shift*	0.085	Sangat Baik
4	*Speed Index*	3.4 detik	Sangat Baik
5	*Total Blocking Time*	0 md	Sangat Baik

V. KESIMPULAN

Perkembangan teknologi salah satunya ialah *smart building* yang merupakan suatu inovasi terbarukan dan dapat digunakan serta diaplikasikan untuk berbagai macam tipe bangunan. Pentingnya teknologi seperti IoT (*Internet of Things*) dibutuhkan dalam pembuatan bangunan menjadi *smart building* untuk menghindari pemborosan energi yang secara berlebihan, membangun strategi manajemen untuk *monitoring* yang efektif. Pemantauan energi pada bangunan sangat dibutuhkan untuk mengetahui nilai seperti suhu atau kelembaban jika mengaktifkan AC (*Air Conditioner*).

Suhu dan kelembaban akan ditampilkan secara *real-time* di *website monitoring* atau *dashboard online* berupa grafik *chart* yang menunjukkan hasil data sensor dan waktu yang dikumpulkan. Untuk membuat *website monitoring* menggunakan bahasa pemrograman Python dengan *framework* Flask yang didukung oleh bahasa pemrograman JavaScript untuk membuat grafik *chart* serta menggunakan MySQL pada XAMPP *Controller* untuk *database* suhu dan kelembaban. Untuk tampilan *website* menggunakan *template* CSS Bootstrap yang dimodifikasi sesuai kebutuhan dengan menggunakan *framework* Flask. Pada sesi pengujian pertama bahwa *website* yang telah dibuat sudah sesuai dengan kriteria yang diinginkan dengan mencoba keseluruhan fitur. Kemudian pengujian secara QoS menggunakan perangkat lunak *Apache* Jmeter untuk menguji performa dari *website monitoring*. Dengan menggunakan sample rentang 10 *users* hingga 1000 *users* untuk mengetahui *throughput*, *delay*, dan *packet loss* sehingga mendapatkan nilai dan indeks yang sesuai dengan indikasi bahwa *website* tersebut dapat memberikan kualitas dan kebutuhan yang diinginkan oleh pengguna yang akan digunakan sebagai *monitoring* suhu dan kelembaban. Akan tetapi pada rentang 500 *users* hingga 1000 *users* mendapatkan nilai tidak bagus pada *delay* dan *packet loss*. Pada pengujian terakhir dengan menggunakan *Page Speed Insight* di mana mendapatkan beberapa indeks penilaian dengan metode *user desktop* dan *user mobile*. Nilai yang didapat mengindikasi bahwa *website* yang telah dibuat sudah mendapatkan nilai yang maksimal pada penilaian *user desktop* dengan performa, aksesibilitas, dan kepraktisan. Akan tetapi, pada penilaian secara *user mobile* mendapatkan nilai minimal dikarenakan tampilan atau gambar yang dipakai yaitu JPG bukan menggunakan AVIF. Pada penilaian *user mobile* nilai aksesibilitas mendapatkan nilai yang maksimal dikarenakan akses untuk digunakan di *mobile* cukup mudah dan gampang.

REFERENSI

[1] D. A. Megawaty, "Sistem Monitoring kegiatan Akademik Siswa menggunakan website," *Jurnal Tekno Kompak*, vol. 14, no. 2, p. 98, 2020. doi:10.33365/jtk.v14i2.756

[2] A. H. Buckman, M. Mayfield, and S. B.M. Beck, "What is a smart building?," *Smart and Sustainable Built Environment*, vol. 3, no. 2, pp. 92–109, 2014. doi:10.1108/sasbe-01-2014-0003

[3] R. L. Jana, S. Dey, and P. Dasgupta, "A hierarchical HVAC control scheme for energy-aware Smart Building Automation," *ACM Transactions on Design Automation of Electronic Systems*, vol. 25, no. 4, pp. 1–33, 2020. doi:10.1145/3393666

[4] A. M. Shalaby *et al.*, "A prototype model of monitoring energy consumption and optimizing distribution of Smart Buildings," *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, 2022. doi:10.1109/iicaiet55139.2022.9936774

[5] L. Alfat, A. Triwiyatno, and R. R. Isnanto, "Sentinel Web: Implementation of Laravel framework in web based temperature and humidity monitoring system," *2015 2nd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, 2015. doi:10.1109/icitacee.2015.7437768

[6] D. Ashley, "Comparing CGI, SSI, Flask, and Django," *Foundation Dynamic Web Pages with Python*, pp. 201–208, 2020. doi:10.1007/978-1-4842-6339-6_7

[7] C. Yuan, "Exploration on interesting and skill programming of web software development," *DEStech Transactions on Social Science, Education and Human Science*,no.ssme,2017.doi:10.12783/dtssehs/ssme2017/1305

[8] C. K. Metallidou, K. E. Psannis, and E. A. Egyptiadou, "Energy efficiency in smart buildings: Iot approaches," *IEEE Access*, vol. 8, pp. 63679–63699, 2020. doi:10.1109/access.2020.2984461

[9] P. H. Pramitasari and S. T. Harjanto, "Building code terhadap pengendalian Kelembaban Bangunan Gedung di Indonesia," *Prosiding SEMSINA*, vol. 3, no. 2, pp. 213–218, 2022. doi:10.36040/semsina.v3i2.5119

[10] R. Abbas, Z. Sultan, and S. N. Bhatti, "Comparative analysis of Automated Load Testing Tools: Apache jmeter, Microsoft Visual Studio (TFS), LoadRunner, siege," *2017 International Conference on Communication Technologies (ComTech)*, 2017. doi:10.1109/comtech.2017.8065747

[11] M. R. Mufid, A. Basofi, M. U. Al Rasyid, I. F. Rochimansyah, and A. rokhim, "Design an MVC model using Python for Flask Framework Development," *2019 International Electronics Symposium (IES)*, 2019. doi:10.1109/elecsym.2019.8901656

[12] S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to build high-performance network programs," *IEEE Internet Computing*, vol. 14, no. 6, pp. 80–83, 2010. doi:10.1109/mic.2010.145

[13] S. Suraya and M. Sholeh, "Designing and implementing a database for thesis data management by using the python flask framework," *International Journal of Engineering, Science and Information Technology*, vol. 2, no. 1, pp. 9–14, 2021. doi:10.52088/ijesty.v2i1.197

[14] R. Oktrifianto, D. Adhipta, and W. Najib, "Page load time speed increase on Disease Outbreak Investigation Information System website," *IJITEE (International Journal of Information Technology and Electrical Engineering)*, vol. 2, no. 4, p. 114, 2019. doi:10.22146/ijitee.46599