Edu Elektrika Journal Vol 12 No 2
Juli – Desember 2024

E-ISSN 2723-5602
P-ISSN 2252-7095

https://journal.unnes.ac.id/journals/eduel/index

Integrasi Cron Job untuk Otomatisasi Pengolahan dan Transfer Data pada Sistem Cloud-Fog

Apriansyah Wibowo[1*], Aisya Fathimah[2], Rizky Ajie Aprilianto[3], dan Deswal Waskito[4]

[1,2,3,4]*Universitas Negeri Semarang*
Jl. Taman Siswa No.83, Sekaran, Kec. Gn. Pati, Kota Semarang, Jawa Tengah, Indonesia
sultanapri@students.unnes.ac.id[1], fathimahaisya1@students.unnes.ac.id[2], rizkyajiea@mail.unnes.ac.id[3], deswalwaskito@students.unnes.ac.id[4]*

Abstrak— Perkembangan pesat perangkat Internet of Things (IoT) telah menghasilkan peningkatan signifikan dalam volume data, yang menciptakan tantangan dalam pemrosesan data secara *real-time*. Meskipun sistem cloud-fog dapat mengurangi latensi dengan memproses data lebih dekat ke sumbernya, pengelolaan dan transfer data tetap bergantung pada tugas terjadwal yang efisien. Penelitian ini bertujuan untuk mengurangi intervensi manual dan meningkatkan efisiensi sistem melalui penggunaan *cron job* yang dikonfigurasi dengan logika skrip PHP dan diimplementasikan di Cloud Panel. Sistem ini dirancang untuk menjadwalkan transfer data secara otomatis dengan interval yang disesuaikan dan memastikan pembaruan data yang tepat waktu dalam lingkungan *cloud-fog*. Hasil penelitian menunjukkan bahwa penerapan otomatisasi transfer data berhasil mengurangi kesalahan, meningkatkan efisiensi, dan memastikan pembaruan data yang lebih tepat waktu dalam sistem *cloud-fog*.

Kata kunci— ***Internet of Things* (IoT), *Cloud-fog*, *Real-time*, PHP, *Cron job*.**

Abstract— The rapid development of Internet of Things (IoT) devices has led to a significant increase in data volume, creating challenges in real-time data processing. While the cloud-fog system can reduce latency by processing data closer to the source, data management and transfer still rely on efficient scheduled tasks. This research aims to reduce manual intervention and improve system efficiency through the use of cron jobs configured with PHP script logic and implemented on the Cloud Panel. The system is designed to automatically schedule data transfer at customized intervals and ensuring timely data updates in the cloud-fog environment. The results of the study show that the implementation of automated data transfer successfully reduces errors, enhances efficiency, and ensures more timely data updates in the cloud-fog system.

Keywords— Internet of Things (IoT), Cloud-fog, Real-time, PHP, Cron job.

I. PENDAHULUAN

Integrasi perangkat *Internet of Things* (IoT) yang terus berkembang di berbagai sektor industri telah menyebabkan peningkatan eksponensial dalam jumlah data yang dihasilkan (Zikria et al., 2021),(Munirathinam, 2020). Seiring dengan upaya berbagai industri untuk memproses data secara *real-time*, meskipun model komputasi awan tradisional sudah cukup untuk penyimpanan dan pemrosesan data dalam jumlah besar (Cheng et al., 2018), tantangan terkait latensi dan kebutuhan pembaruan data yang cepat menjadi poin utama, terutama pada aplikasi yang memerlukan pengambilan keputusan cepat berdasarkan analisis data yang selalu diperbarui (Lu et al., 2020; Shukla et al., 2023). Dalam konteks ini, dibutuhkan sistem yang efisien dan otomatis untuk menangani transfer data menjadi sangat penting (Ariyaluran et al., 2019; Bakken et al., 2011). Salah satu solusi yang dapat diterapkan adalah c*ron job* yang mengotomatisasi operasi berulang pada interval yang telah ditentukan sebelumnya sehingga manajemen data dapat dilakukan dengan tepat waktu dan konsisten (Bourne & Fox, 1984).

Meskipun *cloud data center* telah lama menjadi infrastruktur utama untuk memproses data IoT, ketergantungan pada server jarak jauh serta latensi jaringan memberikan dampak signifikan terhadap kinerja, terutama pada aplikasi yang sensitif terhadap waktu (Bilal et al., 2018).Untuk mengatasi masalah ini, paradigma komputasi terdesentralisasi seperti *edge computing* dan *fog computing* telah dikembangkan sehingga memungkinkan pemrosesan data lebih dekat ke sumbernya guna mengurangi latensi (Hong & Varghese, 2020; Ren et al., 2020). Meskipun pemrosesan data kini lebih dekat ke sumbernya, penggunaan *cron job* yang efisien tetap penting untuk mengoptimalkan penggunaan sumber daya dan memastikan pemrosesan data dilakukan secara *real-time*, tanpa keterlambatan yang dapat memengaruhi kinerja sistem secara keseluruhan (García-Valls et al., 2014; Ngcobo et al., 2024; Tantalaki et al., 2020). Dalam konteks ini, *cron job* digunakan untuk menjadwalkan tugas secara otomatis dan mengelola lokasi terbaik untuk pelaksanaannya, apakah di *node cloud* atau *node fog* sehingga pemrosesan data dapat dilakukan secara efisien dan memenuhi kebutuhan *real time* dengan memanfaatkan sumber daya komputasi yang tersedia di

Diterima: 23 Desember 2024 | Disetujui: 9 Januari 2025 | Dipublikasi: 13 Januari 2025

E-ISSN 2723-5602
P-ISSN 2252-7095
https://journal.unnes.ac.id/journals/eduel/index

masing-masing *node* (Alsadie, 2024; Jamil et al., 2022; Kaur et al., 2023).

Penelitian ini bertujuan untuk mengeksplorasi penerapan *cron job* sebagai solusi untuk mengotomatiskan transfer data antar sistem IoT dalam lingkungan komputasi *cloud-fog*. Fokus utama penelitian ini adalah otomatisasi transfer data di *database* antara dua tabel *MySQL data_now*, yang menyimpan data *real time*, dan *data_one_hour*, yang menyimpan data yang diperbarui setiap jam. Dengan mengintegrasikan *cron job*, diharapkan dapat mengurangi intervensi manual, memastikan pembaruan data yang tepat waktu, serta meningkatkan efisiensi system (Nelly et al., 2024). Selain itu, penelitian ini juga berupaya mengoptimalkan konsumsi energi dengan mengurangi pemrosesan data yang berkelanjutan, sehingga membantu mengatasi tantangan lingkungan dan keuangan terkait penerapan IoT skala besar.

Hasil dari penelitian ini berupa penerapan *cron job* yang efektif dalam model komputasi *cloud-fog* yang dapat secara signifikan meningkatkan efisiensi dan keandalan operasi transfer data. Dengan mengotomatiskan tugas manajemen data yang krusial, *cron job* tidak hanya mengurangi biaya operasional dan kesalahan sistem, tetapi juga mendukung keberlanjutan dan skalabilitas sistem IoT (Chegini et al., 2021). Penelitian ini memberikan analisis mendalam mengenai implementasi, tantangan, dan hasil penggunaan *cron job* untuk otomatisasi distribusi data *real time*, yang menawarkan solusi yang kuat bagi industri yang membutuhkan akurasi dan ketepatan waktu data yang tinggi.

II. Literature Review

Penggunaan *cron job* dalam sistem terdistribusi, terutama yang berbasis *cloud* dan *fog computing*, telah menjadi topik penelitian yang menarik dalam beberapa tahun terakhir (Alizadeh et al., 2020). Penelitian sebelumnya telah mengidentifikasi berbagai teknik untuk mengoptimalkan penggunaan *cron job*, dengan fokus pada efisiensi waktu dan pemanfaatan sumber daya. Salah satu pendekatan yang signifikan adalah penerapan algoritma optimasi, seperti yang dijelaskan dalam (Satouf et al., 2025), yang menyarankan penggunaan *Grey Wolf Optimizer* (GWO) untuk penjadwalan *real-time*. Pendekatan ini berhasil meningkatkan kinerja dalam beberapa aspek, termasuk waktu eksekusi, biaya, dan konsumsi energi, dengan mempertimbangkan durasi tugas serta kebutuhan sumber daya. Meskipun efektif, penerapan GWO memerlukan sumber daya komputasi yang cukup besar, yang dapat membatasi efektivitasnya dalam lingkungan yang sangat dinamis dan terdistribusi, seperti sistem *cloud* yang sering mengalami fluktuasi sumber daya secara *real-time*.

Dalam konteks simulasi penjadwalan, Ref. (Dirdal et al., 2024) mengembangkan sebuah kerangka kerja simulasi berbasis jaringan Petri untuk memodelkan algoritma penjadwalan multiprosesor. Kerangka kerja ini menunjukkan kemampuan jaringan Petri dalam merepresentasikan proses bersamaan dan interaksi dinamis, serta menganalisis metrik kinerja seperti waktu penyelesaian dan *throughput*. Penelitian ini juga membahas penggunaan algoritma penjadwalan tradisional, seperti *First-Come-First-Served* (FCFS) dan *Shortest Job First* (SJF), yang memiliki karakteristik tertentu dalam menangani berbagai beban kerja. Meskipun demikian, asumsi model, seperti penyederhanaan operasi I/O dalam jaringan Petri, dapat membatasi akurasi penerapannya, terutama dalam lingkungan interaktif atau sistem *real-time*, di mana faktor-faktor dinamis lebih kompleks dan harus dipertimbangkan lebih rinci.

Dalam domain otomatisasi penjadwalan, studi oleh (Lyo et al., 2021) mengusulkan sistem *RadRemind* untuk otomatisasi pengingat tugas, yang bertujuan untuk meningkatkan manajemen tugas di kalangan residen radiologi. Sistem ini, yang dibangun menggunakan *PHP*, *MySQL*, dan *Cron Job*, menunjukkan efektivitasnya dalam mengurangi tugas yang terlewat dan kecemasan peserta. Hasil penelitian menunjukkan bahwa sistem ini berhasil meningkatkan tingkat ingatan dan mengurangi kecemasan, yang mengindikasikan bahwa otomatisasi melalui *cron job* dapat membawa manfaat signifikan dalam pengelolaan tugas secara efisien. Namun demikian, faktor eksternal, seperti migrasi server dan bias seleksi dalam respons survei, dapat mempengaruhi kemampuan sistem untuk digeneralisasi ke aplikasi lainnya, sehingga penelitian lebih lanjut diperlukan untuk mengatasi tantangan ini dan memperluas penerapan sistem ke domain lain.

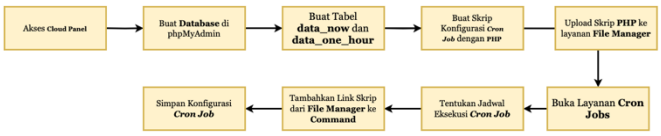
Penelitian yang ada telah banyak membahas aspek-aspek teknis penjadwalan dan otomatisasi tugas, namun masih ada kesenjangan yang perlu diperbaiki. Banyak penelitian lebih berfokus pada pengoptimalan algoritma individual atau pengembangan alat simulasi tanpa mengintegrasikan pendekatan tersebut dalam sistem adaptif yang dapat berjalan secara otomatis dalam lingkungan dinamis dan *real time*. Selain itu, tantangan dalam *overhead* komputasi dan asumsi model pada penelitian sebelumnya menunjukkan perlunya solusi yang lebih terukur dan tangguh, yang dapat diterapkan dalam sistem *cloud-fog* dengan fluktuasi sumber daya yang tinggi.

Sebagai solusi terhadap kesenjangan yang ada, penelitian ini mengusulkan pengembangan platform yang mengintegrasikan cron job otomatis dengan kemampuan simulasi dinamis untuk distribusi data. Platform ini memanfaatkan *cron job* untuk menjadwalkan transfer data otomatis dengan interval waktu yang disesuaikan, serta menggunakan pengoptimalan tingkat lanjut untuk memastikan sistem dapat berjalan efisien dan efektif dalam lingkungan *cloud*. Pendekatan ini menggabungkan kemajuan yang ditemukan dalam literatur sebelumnya, menawarkan solusi praktis untuk pengelolaan tugas dan distribusi data secara otomatis dengan pengurangan beban kerja manual dan peningkatan produktivitas.

III. Metode Penelitian

Penelitian ini menggunakan pendekatan sistematis untuk mengintegrasikan cron job dalam pengelolaan data secara otomatis di Cloud Panel. Cloud Panel berfungsi sebagai platform untuk konfigurasi dan penjadwalan *cron job*, sementara skrip PHP digunakan untuk mengatur logika transfer data dan memastikan proses berjalan sesuai jadwal. Sub-bagian berikut menjelaskan langkah-langkah terperinci dari setiap tahapan proses tersebut.

E-ISSN 2723-5602
P-ISSN 2252-7095
https://journal.unnes.ac.id/journals/eduel/index



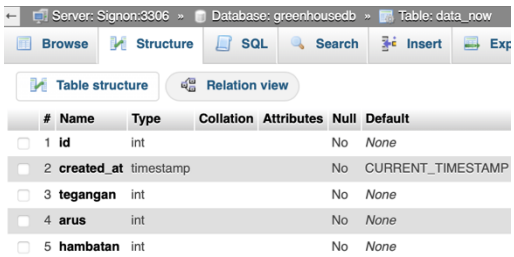
Gambar 1. Diagram blok integrasi *cron job*

A. Akses Cloud Panel dan Pembuatan *Database*

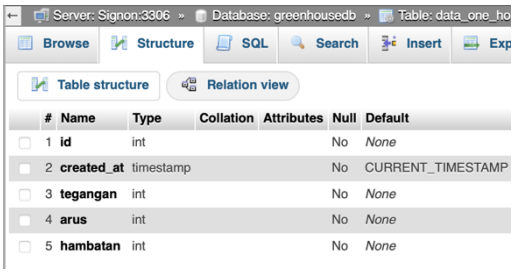
Langkah pertama dalam pengembangan sistem adalah mengakses Cloud Panel untuk membuat *database* melalui phpMyAdmin sebagai tempat penyimpanan data. Dua tabel utama dibuat, yaitu *data_now* untuk menyimpan data real-time dan *data_one_hour* untuk menyimpan hasil transfer data dengan interval waktu satu jam. Struktur tabel ini dirancang untuk mendukung pengolahan dan penyimpanan data yang optimal sesuai kebutuhan distribusi otomatis. Proses pembuatan database dilakukan terlebih dahulu di phpMyAdmin, seperti yang ditunjukkan pada Gambar 2.

Gambar 2. Buat *database* di phpMyAdmin

Setelah *database* dibuat, dua tabel utama *data_now* dan *data_one_hour* ditambahkan ke dalamnya. Desain dan struktur tabel tersebut dapat dilihat pada Gambar 3 dan Gambar 4.



Gambar 3. Tabel *data_now*



Gambar 4. Tabel *data_one_hour*

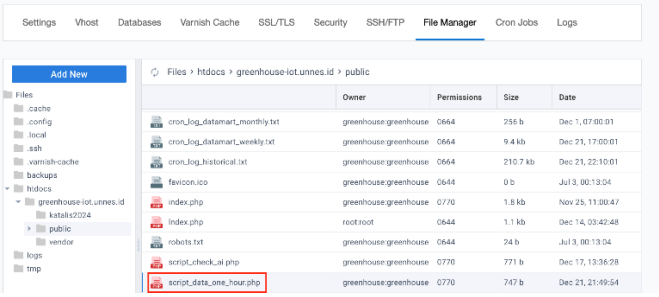
B. Pembuatan dan Penambahan Skrip PHP ke *File Manager*

Setelah database selesai dikonfigurasi, langkah berikutnya adalah menyusun skrip PHP untuk mengelola proses transfer data secara otomatis. Automasi data dilakukan dengan menentukan jadwal eksekusi yang memastikan data ditransfer secara teratur sesuai waktu yang telah ditentukan (Tarantilis et al., 2008). Skrip ini mencakup logika pemrosesan transfer data dari tabel *data_now* ke tabel *data_one_hour*, termasuk pengaturan waktu eksekusi menggunakan *cron job*.

```php
<?php
$conn = new mysqli('127.0.0.1', 'greenhouse', 'niFurVVaTtmW53egIGmu', 'greenhousedb');

if ($conn->connect_error) {
    die("Koneksi gagal: " . $conn->connect_error);
}

$sql_select_data_one_hour = "
    INSERT INTO data_one_hour (created_at, tegangan, arus, hambatan)
    SELECT created_at, tegangan, arus, hambatan
    WHERE created_at >= NOW() - INTERVAL '1 hour';
";

$conn->close();
?>
```

Gambar 5. Penambahan skrip PHP ke *File Manager*

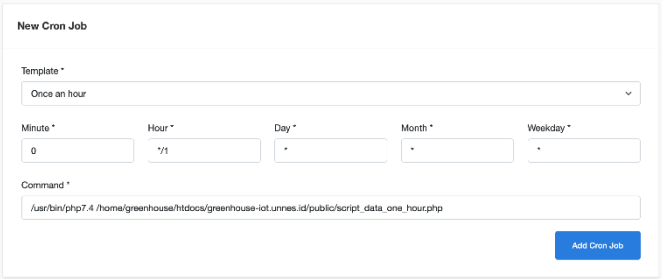
Skrip ini mengotomatisasi proses pemindahan data dari tabel *real-time* ke tabel penyimpanan berdasarkan interval waktu yang ditentukan, sehingga mendukung distribusi data terjadwal. Kemudian, tempatkan skrip PHP yang telah dibuat ke direktori File Manager seperti Gambar 6.



Gambar 6. Penambahan skrip PHP ke *File Manager*

C. Konfigurasi *Cron Job*

Selanjutnya, implementasi *cron job* pada layanan Cloud Panel dirancang untuk mengotomatisasi tugas berdasarkan jadwal waktu tertentu, memungkinkan transfer data secara otomatis sesuai dengan interval yang telah ditetapkan (Choi, 2024). Dalam konteks penelitian ini, cron job diatur untuk mengeksekusi transfer data *real-time* dari tabel *data_now* ke tabel *data_one_hour* setiap satu jam sekali. Interval tersebut dipilih untuk memenuhi kebutuhan distribusi data yang terjadwal secara efisien. Pengaturan ini memastikan proses transfer data dilakukan secara konsisten dan otomatis pada waktu yang telah ditentukan.



D. Penambahan *Cron Job*

Cron job telah ditambahkan dan dikonfigurasi pada server dalam lingkungan *cloud* untuk memastikan eksekusi tugas berjalan secara otomatis sesuai interval yang telah dijadwalkan.

E-ISSN 2723-5602
P-ISSN 2252-7095
https://journal.unnes.ac.id/journals/eduel/index

Perintah *cron job* yang diintegrasikan melalui antarmuka Cloud Panel, memungkinkan koneksi langsung dengan server cloud untuk mendukung otomatisasi proses secara efisien.

Gambar 8. *Cron job* sukses ditambahka

Setelah konfigurasi selesai, *cron job* akan aktif dan melaksanakan tugas secara otomatis sesuai konfigurasi yang ditambahkan.

IV. HASIL DAN PEMBAHASAN

Hasil penerapan pengotomatisasian distribusi data dengan *cron job* dijabarkan pada bagian ini untuk mengevaluasi efektivitas dan efisiensi sistem yang telah dikembangkan. Dengan konfigurasi *cron job* yang sudah diimplementasikan, pemantauan terkait hasil dapat dilihat pada tabel *database* yang telah dibuat. Dua tabel MySQL yang digunakan meliputi data *real-time* dari *data_now* dan tabel *data_one_hour* yang menerima data setiap satu jam sekali dari tabel *data_now*.

id	created_at	tegangan	arus	hambatan
1	2024-12-09 00:00:00	221	10	22
2	2024-12-09 00:00:05	221	10	22
3	2024-12-09 00:00:10	220	10	21
4	2024-12-09 00:00:15	220	10	22
5	2024-12-09 00:00:20	222	10	21
6	2024-12-09 00:00:25	220	10	22
7	2024-12-09 00:00:30	222	11	21
8	2024-12-09 00:00:35	221	10	21
9	2024-12-09 00:00:40	221	10	21
10	2024-12-09 00:00:45	220	10	22
11	2024-12-09 00:00:50	221	10	22
12	2024-12-09 00:00:55	221	10	22
13	2024-12-09 00:01:00	220	10	21

Gambar 9. Tabel SQL data_now

Pada tabel *data_now*, ditampilkan data dengan pembaruan secara *real-time*, di mana setiap entri data direkam per detik. Kebutuhan utama dalam penelitian ini adalah menjadwalkan distribusi data secara otomatis setiap 1 jam sekali dari *data_now* ke tabel *data_one_hour* menggunakan *cron job*. *Cron job* memastikan bahwa data yang dipindahkan mewakili kondisi terbaru pada waktu eksekusi yang telah dijadwalkan, sehingga sistem selalu mencerminkan keadaan data terkini dalam interval yang lebih luas.

id	created_at	tegangan	arus	hambatan
1	2024-12-09 00:00:00	221	10	22
2	2024-12-09 01:00:00	221	10	22
3	2024-12-09 02:00:00	220	10	21
4	2024-12-09 03:00:00	220	10	22
5	2024-12-09 04:00:00	222	10	21
6	2024-12-09 05:00:00	220	10	22
7	2024-12-09 06:00:00	222	11	21
8	2024-12-09 07:00:00	221	10	21
9	2024-12-09 08:00:00	221	10	21
10	2024-12-09 09:00:00	220	10	22
11	2024-12-09 10:00:00	221	10	22
12	2024-12-09 11:00:00	221	10	22
13	2024-12-09 12:00:00	220	10	21

Gambar 10. Tabel SQL data_one_hour

Pada tabel *data_one_hour*, dapat dilihat bahwa otomatisasi pengolahan data dari *data_now* telah berjalan dengan sukses. Data pada tabel ini mencatat informasi yang diperbarui secara otomatis setiap satu jam sekali, sebagaimana ditunjukkan oleh kolom tanggal dan waktu yang terlampir. Ini membuktikan bahwa proses distribusi dan pemrosesan data berjalan sesuai perencanaan tanpa adanya kesalahan. Automasi ini menunjukkan bahwa pendekatan cron job dapat diandalkan untuk mendukung kebutuhan sistem yang memerlukan konsistensi dan akurasi tinggi.

Hasil ini menunjukkan bahwa langkah-langkah metodologi yang diterapkan, seperti konfigurasi database, penyusunan skrip PHP, dan penjadwalan cron job di Cloud Panel, telah berjalan sesuai rencana. Implementasi ini memberikan kontribusi yang signifikan dalam meminimalkan beban kerja manual, meningkatkan efisiensi distribusi data, dan memastikan konsistensi data real-time. Pendekatan ini dapat diadaptasi untuk berbagai sistem lain yang membutuhkan pengelolaan data terjadwal secara otomatis. Dengan demikian, Hasil ini tidak hanya menunjukkan keberhasilan penelitian tetapi juga memberikan peluang untuk pengembangan lebih lanjut dalam mengintegrasikan sistem serupa pada skala yang lebih besar.

V. PENUTUP

Penelitian ini menyoroti peran penting penggunaan *cron job* dalam mengoptimalkan proses transfer data pada lingkungan *cloud-fog*, khususnya untuk sistem *Internet of Things* (IoT) yang memerlukan manajemen data yang efisien. Dengan otomatisasi transfer data melalui *cron job*, hal ini menawarkan solusi praktis dan efisien untuk mengelola operasi data berskala besar yang sensitif terhadap waktu. Hasil penelitian ini menunjukkan bahwa integrasi *cron job* secara signifikan dapat meningkatkan konsistensi data, mengurangi kesalahan operasional, dan mendukung skalabilitas sistem. Selain itu, dengan mengotomatisasi tugas-tugas yang sebelumnya dilakukan secara manual, sistem ini memastikan pembaruan data dapat berlangsung tanpa penundaan, sehingga dapat memenuhi tuntutan ketat pada aplikasi *real-time*.

Hasil penelitian ini tidak hanya menunjukkan kelayakan teknis dari otomatisasi *cron job*, tetapi juga menekankan manfaat dari sisi lingkungan dan finansial. Dengan mengurangi pemrosesan data secara terus-menerus, konsumsi energi dapat dioptimalkan, yang turut mendukung keberlanjutan sistem IoT. Hal ini menjadi sangat penting dalam konteks penerapan IoT berskala besar, di mana konsumsi sumber daya merupakan salah satu perhatian utama.

Penelitian lebih lanjut dapat mengeksplorasi pengoptimalan penjadwalan cron job dengan algoritma canggih, seperti *machine learning* atau kecerdasan buatan, untuk dapat lebih efektif beradaptasi dengan lingkungan IoT yang dinamis. Selain itu, integrasi *edge computing* dan *fog* dengan *cron job* juga dapat diperluas untuk menyediakan solusi yang lebih tangguh bagi aplikasi-aplikasi yang sangat kritis dan terbatas dalam sumber daya. Dengan demikian, penelitian ini memberikan dasar yang kuat untuk pengembangan dan penyempurnaan sistem otomatisasi yang berkelanjutan dalam

E-ISSN 2723-5602
P-ISSN 2252-7095
https://journal.unnes.ac.id/journals/eduel/index

ekosistem IoT, yang membuka jalan bagi solusi yang lebih efisien, berkelanjutan, dan andal di masa mendatang.

Referensi

Alizadeh, M. R., Khajehvand, V., Rahmani, A. M., & Akbari, E. (2020). Task scheduling approaches in fog computing: A systematic review. International Journal of Communication Systems, 33(16). https://doi.org/10.1002/dac.4583

Alsadie, D. (2024). Advancements in heuristic task scheduling for IoT applications in fog-cloud computing: challenges and prospects. PeerJ Computer Science, 10, e2128. https://doi.org/10.7717/peerj-cs.2128

Ariyaluran Habeeb, R. A., Nasaruddin, F., Gani, A., Targio Hashem, I. A., Ahmed, E., & Imran, M. (2019). Real-time big data processing for anomaly detection: A Survey. International Journal of Information Management, 45, 289–307. https://doi.org/10.1016/j.ijinfomgt.2018.08.006

Bakken, D. E., Bose, A., Hauser, C. H., Whitehead, D. E., & Zweigle, G. C. (2011). Smart Generation and Transmission With Coherent, Real-Time Data. Proceedings of the IEEE, 99(6), 928–951. https://doi.org/10.1109/JPROC.2011.2116110

Bilal, K., Khalid, O., Erbad, A., & Khan, S. U. (2018). Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers. Computer Networks, 130, 94–120. https://doi.org/10.1016/j.comnet.2017.10.002

Bourne, & Fox. (1984). Autonomous Manufacturing: Automating the Job-Shop. Computer, 17(9), 76–86. https://doi.org/10.1109/MC.1984.1659248

Chegini, H., Naha, R. K., Mahanti, A., & Thulasiraman, P. (2021). Process Automation in an IoT–Fog–Cloud Ecosystem: A Survey and Taxonomy. IoT, 2(1), 92–118. https://doi.org/10.3390/iot2010006

Cheng, B., Zhang, J., Hancke, G. P., Karnouskos, S., & Colombo, A. W. (2018). Industrial Cyberphysical Systems: Realizing Cloud-Based Big Data Infrastructures. IEEE Industrial Electronics Magazine, 12(1), 25–35. https://doi.org/10.1109/MIE.2017.2788850

Choi, B. (2024). Python Network Automation Labs cron. In Introduction to Python Network Automation Volume II (pp. 229–271). Apress. https://doi.org/10.1007/979-8-8688-0391-8_4

Dirdal, D. O., Vo, D., Feng, Y., & Davidrajuh, R. (2024). Developing a Platform Using Petri Nets and GPenSIM for Simulation of Multiprocessor Scheduling Algorithms. Applied Sciences, 14(13), 5690. https://doi.org/10.3390/app14135690

García-Valls, M., Cucinotta, T., & Lu, C. (2014). Challenges in real-time virtualization and predictable cloud computing. Journal of Systems Architecture, 60(9), 726–740. https://doi.org/10.1016/j.sysarc.2014.07.004

Hong, C.-H., & Varghese, B. (2020). Resource Management in Fog/Edge Computing. ACM Computing Surveys, 52(5), 1–37. https://doi.org/10.1145/3326066

Jamil, B., Ijaz, H., Shojafar, M., Munir, K., & Buyya, R. (2022). Resource Allocation and Task Scheduling in Fog Computing and Internet of Everything Environments: A Taxonomy, Review, and Future Directions. ACM Computing Surveys, 54(11s), 1–38. https://doi.org/10.1145/3513002

Kaur, A., Auluck, N., & Rana, O. (2023). Real-Time Scheduling on Hierarchical Heterogeneous Fog Networks. IEEE Transactions on Services Computing, 16(2), 1358–1372. https://doi.org/10.1109/TSC.2022.3155783

Lu, Y., Liu, L., Panneerselvam, J., Zhai, X., Sun, X., & Antonopoulos, N. (2020). Latency-Based Analytic Approach to Forecast Cloud Workload Trend for Sustainable Datacenters. IEEE Transactions on Sustainable Computing, 5(3), 308–318. https://doi.org/10.1109/TSUSC.2019.2905728

Lyo, S., Grigorian, A., Cohen-Addad, D., & Kolla, S. (2021). RadRemind!—an Automated Paging System for Reminding Radiology Residents of Their Roles and Responsibilities. Journal of Digital Imaging, 34(2), 489–494. https://doi.org/10.1007/s10278-021-00437-2

Munirathinam, S. (2020). Industry 4.0: Industrial Internet of Things (IIOT) (pp. 129–164). https://doi.org/10.1016/bs.adcom.2019.10.010

Nelly Tochi Nwosu. (2024). Reducing operational costs in healthcare through advanced BI tools and data integration. World Journal of Advanced Research and Reviews, 22(3), 1144–1156. https://doi.org/10.30574/wjarr.2024.22.3.1774

Ngcobo, K., Bhengu, S., Mudau, A., Thango (Y2-rated Researcher), B., & Matshaka, L. (2024). From Single Shot to Structure: End-to-End Network based Deflectometry for Specular Free-Form Surface Reconstruction. https://doi.org/10.20944/preprints202409.1913.v1

Ren, J., Zhang, D., He, S., Zhang, Y., & Li, T. (2020). A Survey on End-Edge-Cloud Orchestrated Network Computing Paradigms. ACM Computing Surveys, 52(6), 1–36. https://doi.org/10.1145/3362031

Satouf, A., Hamidoğlu, A., Gül, Ö. M., Kuusik, A., Durak Ata, L., & Kadry, S. (2025). Metaheuristic-based task scheduling for latency-sensitive IoT applications in edge computing. Cluster Computing, 28(2), 143. https://doi.org/10.1007/s10586-024-04878-6

Shukla, S., Hassan, Mohd. F., Tran, D. C., Akbar, R., Paputungan, I. V., & Khan, M. K. (2023). Improving latency in Internet-of-Things and cloud computing for real-time data transmission: a systematic literature review (SLR). Cluster Computing, 26(5), 2657–2680. https://doi.org/10.1007/s10586-021-03279-3

Tantalaki, N., Souravlas, S., & Roumeliotis, M. (2020). A review on big data real-time stream processing and its scheduling techniques. International Journal of Parallel,

E-ISSN 2723-5602
P-ISSN 2252-7095
https://journal.unnes.ac.id/journals/eduel/index

Emergent and Distributed Systems, 35(5), 571–601. https://doi.org/10.1080/17445760.2019.1585848

Tarantilis, C. D., Kiranoudis, C. T., & Theodorakopoulos, N. D. (2008). A Web-based ERP system for business services and supply chain management: Application to real-world process scheduling. European Journal of Operational Research, 187(3), 1310–1326. https://doi.org/10.1016/j.ejor.2006.09.015

Zikria, Y. Bin, Ali, R., Afzal, M. K., & Kim, S. W. (2021). Next-Generation Internet of Things (IoT): Opportunities, Challenges, and Solutions. Sensors, 21(4), 1174. https://doi.org/10.3390/s21041174

