PENGIMPLEMENTASIAN DEVOPS PADA SISTEM TERTANAM DENGAN ESP8266 MENGGUNAKAN MEKANISME OVER THE AIR

Lili Andraini
Teknik Komputer
*) liliandraini@gmail.com

Abstrak

DevOps mendorong percepatan pengembangan sistem. Namun bukti nyata penerapannya pada sistem tertanam belum mencukupi. Salah satu penyebabnya adalah kesulitan proses deployment pada perangkat. Konsep IoT menghubungkan sistem tertanam dengan jaringan yang memungkinkan proses pembaharuan firmware menggunakan mekanisme Over The Air (OTA). Tulisan ini mengusulkan infrastruktur DevOps untuk pengembangan sistem tertanam. Perangkat keras yang digunakan adalah microcontroller ESP8266. Sedangkan lingkungan DevOps menggunakan perangkat lunak PlatformIO, GitHub dan Travis CI. Pengujian dilakukan dengan mengubah user requirement yang kemudian diterapkan pada perangkat keras. Tahapan DevOps (build and test, release hingga deploy) telah berhasil dilakukan secara otomatis. Sistem mampu mendeteksi kesalahan penulisan kode sumber. Rerata waktu keseluruhan proses adalah 77,21 detik. Proses build and test mendominasi waktu proses dengan rerata sebesar 77,21 detik dan waktu deploy memiliki rerata 1,41 detik.

Kata Kunci: Iot, sistem tertanam,OTA,DevOps,ESP8266

PENDAHULUAN

Perkembangan *Internet of Things* (IoT) akhir-akhir ini cukup pesat dan telah memasuki berbagai macam bidang (Amarudin et al., 2020). Sebagai contoh adalah pengembangan sistem IoT pada bidang energi serta bidang keamanan rumah (Kholidi dkk., 2015), (Subandi, 2016). Perkembangan ini semakin didorong oleh integrasi komputasi *cloud* sehingga memudahkan pengolahan data. Integrasi tersebut memperkenalkan dimensi baru pada *big data* dan analisis data. Penggabungan konsep *big data* dan IoT semakin memudahkan penggalian data serta mendorong pemanfaatan IoT secara massif (Widodo et al., 2020), (Selamet Samsugi & Wajiran, 2020),. IoT dapat didefinisikan sebagai interkoneksi antar piranti sensor maupun aktuator yang memiliki kemampuan untuk membagikan informasi pada lintas *platform* melalui *platform* terpadu guna membangun suatu gambaran operasi bersama yang memungkinkan aplikasi inovatif(S Samsugi et al., 2021), (Rikendry & Navigasi, 2007). Hal ini diperoleh melalui *seamless ubiquitous sensing*, analisis data, serta representasi informasi dengan komputasi *cloud* sebagai *framework* penyatu (Susanto, n.d.), (Yurnama & Azman, 2009). IoT juga menawarkan kemudahan untuk menanamkan kecerdasan pada perangkat yang pada akhirnya dapat berkomunikasi satu sama lain untuk bertukar informasi (Zanofa et al., 2020), (Wantoro et al., 2021)

Dalam penerapan IoT, pengembangan sistem merupakan sesuatu yang pasti ada. Sering kali terjadi celah antara pengembangan dan pengoperasian. Pengembang sering kali akan melakukan perubahan kode yang membutuhkan penggabungan berkali-kali bahkan dalam jangka waktu satu hari (Pindrayana et al., 2018), (D. E. Kurniawan et al., 2019). Untuk mengatasi hal tersebut maka dikembangkan konsep *DevOps* yang berarti penggabungan

antara pengembangan dan operasional guna menyederhanakan proses *delivery* perangkat lunak, memperkuat proses pembelajaran dengan langsung mengarahkan umpan balik dari proses produksi kepada pengembang dan memperbaiki waktu siklus (Borman et al., 2018), (Imani & Ghassemian, 2019),. Dengan perkembangan IoT yang cukup pesat, maka sistem IoT yang dikembangkan juga perlu untuk beradaptasi dengan cepat untuk memenuhi kebutuhan baru. berbagai macam perangkat lunak yang dapat digunakan untuk sistem IoT untuk menjamin *Continuous Integration* (CI) dan *Continuous Deployment* (CD). Untuk menjamin proses *deployment* secara berkesinambungan pada perangkat IoT diperlukan strategi proses pembaharuan *firmware* (Yulianti et al., 2021), (Ahdan et al., 2019). Penerapan *DevOps* pada IoT dapat melibatkan *containerization* menggunakan *docker* untuk mempermudah *deployment* pada perangkat yang banyak, *docker* untuk menerapkan *DevOps* pada sistem informasi yang memiliki kemampuan untuk melakukan analisis waktu nyata penggunaan kendaraan listrik pada *smart city* (Budioko, 2016). Beberapa pengembangan sistem IoT telah mengadopsi *DevOps* dengan memanfaatkan *docker* untuk proses *Continuous Deployment* pada. Penggunaan *docker* ini mengharuskan penggunaan sistem operasi sebagai *host* bagi *docker* yang dijalankan. Untuk memenuhi kebutuhan tersebut banyak digunakan komputer SoC (*System on a Chip*) Raspberry Pi Meskipun demikian, sistem-sistem yang dikembangkan di atas masih menggunakan sistem operasi yang kompleks, dan tidak dapat menggunakan perangkat *microcontroller* langsung (S Samsugi & Silaban, 2018a), (Selamet Samsugi et al., 2018),.

KAJIAN PUSTAKA

IOT (Internet Of Things)

Internet of things merupakan sebuah konsep di mana suatu benda atau objek ditanamkan teknologi-teknologi seperti sensor dan software dengan tujuan untuk berkomunikasi, mengendalikan, menghubungkan, dan bertukar data melalui perangkat lain selama masih terhubung ke internet (Rumalutur & Ohoiwutun, 2018), (Jayadi et al., 2021). Untuk membuat suatu ekosistem IoT, kita tidak hanya memerlukan perangkat-perangkat yang pintar, melainkan juga berbagai unsur pendukung lain di dalamnya (F. Kurniawan & Surahman, 2021),. IoT memiliki hubungan yang erat dengan istilah machine-to-machine atau M2M. Seluruh alat yang memiliki kemampuan komunikasi M2M ini sering disebut dengan perangkat cerdas atau smart devices. Perangkat cerdas ini diharapkan dapat membantu kerja manusia dalam menyelesaikan berbagai urusan atau tugas yang ada. Berikut adalah berbagai unsur pembentuk *internet of things*, terkait kecerdasan buatan, sensor, dan konektivitas , (S Samsugi & Silaban, 2018b), (Nurdiansyah et al., 2020), (Setiawan et al., 2021).

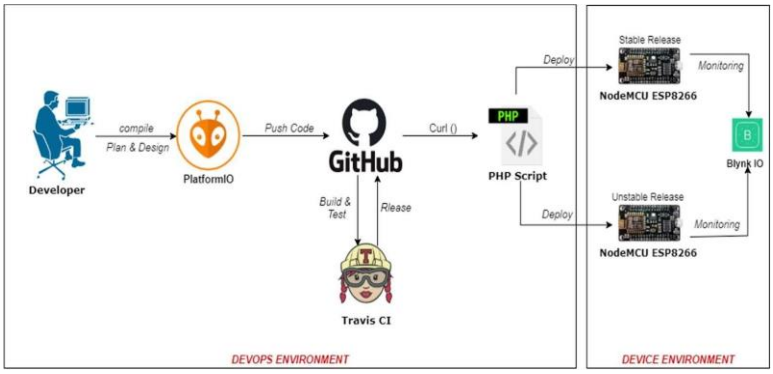
Sistem Tertanam

Sistem Tertanam merupakan sebuah sistem (rangkaian elektronik) digital yang merupakan bagian dari Sebuah sistem yang lebih besar, yang biasanya bukan berupa sistem elektronik. Secara umum, sistem tertanam dirancang untuk aplikasi tertentu (*application specific system*) (Surahman et al., 2021), (Ratnasari et al., n.d.),. Contoh pengaplikasiannya adalah instrumentasi medik, *automated vehicles control*, dan perangkat komunikasi. Berbeda dengan sistem digital yang dirancang untuk general purpose. Umumnya, sistem tertanam diimplementasikan melalui microcontroller. Sistem tertanam memberikan respon secara real time dan banyak digunakan di peralatan digital, salah satu nya yaitu jam tangan (Selamet Samsugi, Yusuf, et al., 2020), (Isnain et al., 2021), (S Samsugi, 2017).

METODE

Rancangan Infrastruktur *DevOps* Sistem Tertanam dengan menggunakan ESP8266

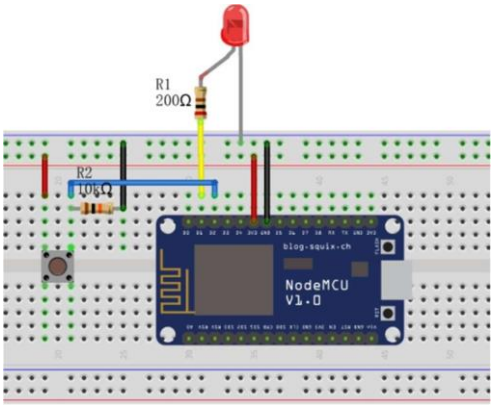
Infrastruktur *DevOps* yang diusulkan dapat dilihat pada Gambar 1. Pada prinsipnya terdapat dua lingkungan, yaitu lingkungan *DevOps* dan lingkungan perangkat keras. Perbedaan dengan konsep *DevOps* perangkat lunak pada umumnya, adalah penggunaan sistem tertanam ESP8266 (Hafidhin et al., 2020), (Selamet Samsugi et al., 2021), (Dita et al., 2021). Pada Gambar 1 terlihat komponen penyusun infrastruktur *DevOps* untuk pengembangan sistem tertanam ESP8266 terdiri atas beberapa peranti dan perangkat sebagai berikut. Lingkungan perangkat keras sistem tertanam dan Lingkungan *DevOps* (Kristiawan et al., 2021), (Riski et al., 2021)



Gambar 1. Rancangan Infrastruktur *DevOps* untuk Sistem Tertanam ESP8266.

Rancangan Perangkat Keras Sistem Tertanam

Perangkat sistem tertanam yang digunakan untuk percobaan cukup sederhana. Terdiri dari sebuah NodeMCU, sebuah *push button* dan sebuah LED. Perangkat keras ini dibuat dua set, satu untuk perangkat keras tes (untuk rilis *firmware unstable*) dan satu untuk perangkat keras operasional (untuk rilis *firmware stable*) (Pratama Zanofa & Fahrizal, 2021), (Yuliana et al., 2021).



Gambar 2. *Wiring Diagram* Perangkat Sistem Tertanam.

Penerapan *DevOps* pada Pengembangan Perangkat Tertanam

Pengembangan sistem dengan menggunakan *DevOps* pada umumnya diterapkan pada piranti tehubung jaringan. Proses *DevOps* sendiri paling tidak terdiri atas tahapan *plan*, *code*, *build*, *test*, *release*, *deploy*, serta *feedback*. Pada simulasi ini akan direncanakan tiga fase pengembangan (Anantama et al., 2020), (Pratama et al., 2021). Rangkuman simulasi perubahan *user requirement* pada tahap *plan* ketiga fase simulasi dapat dilihat pada Tabel 1. Dari Tabel 1 dapat dilihat bahwa *requirement* baru untuk setiap fasenya ditunjukkan

dengan huruf miring dan latar belakang abu-abu. Pada fase pertama LED akan berkedip ketika tombol *push button* ditekan. Sementara fitur pengesetan koneksi wifi melalui peramban dimunculkan pada fase kedua. Pada fase ketiga perilaku LED berubah dari berkedip menjadi berubah keadaannya bergantian dari mati, menyala redup dan menyala terang ketika tombol ditekan (Hayatunnufus & Alita, 2020),. Berdasarkan *user requirement* tersebut, selanjutnya dibuat *use case diagram* seperti yang terlihat pada Tabel 1 dan *class diagram*. Tahap selanjutnya adalah tahap pembuatan kode. Pembuatan kode dilakukan menggunakan *PlatformIO* dengan menggunakan *platform* Espressif8266 dan *framework* Arduino. Penggunaan *PlatformIO* sebagai lingkungan pengembangan karena kemudahan integrasinya dengan GitHub sebagai sistem *versioning* yang memungkinkan kolaborasi dengan mudah dan Travis CI untuk pengujian dan *deplyoment* otomatis (Suaidah, 2021), (Puspaningrum et al., 2020), (S Samsugi & Burlian, 2019).

Penerapan Mekanisme *Firmware Update Over The Air*

Mekanisme pembaharuan *firmware over the air* menggunakan *library* ESP8266 *httpUpdate* (S Samsugi et al., 2018),. Pada prinsipnya perangkat tertanam akan melakukan pengecekan dan pengunduhan *firmware* rilis terbaru dari *repository* GitHub melalui skrip PHP pada *server*. Informasi keberadaan *firmware* versi terbaru dapat dilihat oleh pengguna melalui aplikasi Blynk pada perangkat *smartphone* (Rahmanto et al., 2020), (S Samsugi & Suwantoro, 2018). Pada aplikasi Blynk terdapat tombol untuk menjalankan pembaharuan *firmware* ketika terdapat versi *firmware* yang baru. Ketika pengguna menekan tombol tersebut, maka perangkat tertanam akan menjalankan mekanisme *firmware update over the air* dengan memanggil skrip php pada *server*. Pada prinsipnya perangkat tertanam akan menggunakan *server* dengan skrip PHP tersebut sebagai perantara (Ahmad et al., 2022), (Selamet Samsugi, Mardiyansyah, et al., 2020), (Rahmanto et al., 2021),.

HASIL DAN PEMBAHASAN

Automatic Test pada Pengembangan

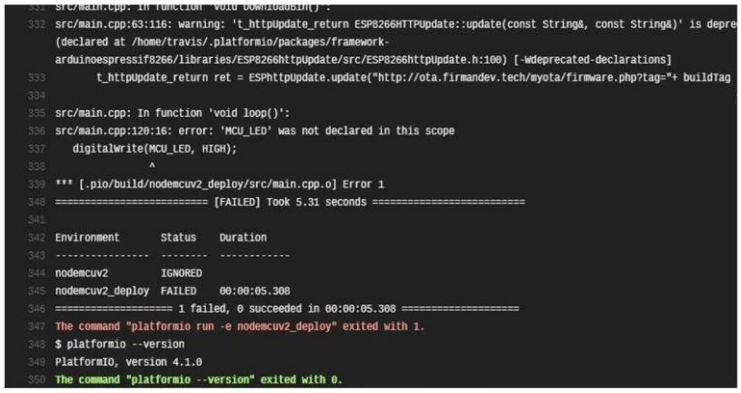
Automatic test adalah pengujian yang dilakukan untuk menguji keberhasilan automasi tahapan proses sistem. Tahapan proses yang diuji dimulai dari tahap pengembang membuat kode sumber yang dilanjutkan dengan mendorong kode baru tersebut ke GitHub hingga tahap *deploymet* ke perangkat ESP8266. Uji ini dilakukan sebanyak tujuh kali dengan melakukan modifikasi terhadap kode sumber.

Dari ketujuh uji tersebut, terdapat dua kali simulasi kegagalan dengan melakukan kesalahan pada penulisan kode sumber. Tabel 2 menunjukkan hasil dari ketujuh uji yang dilakukan. Baris pertama, kedua, keempat, kelima dan keenam menunjukkan keberhasilan proses, sedangkan baris ketiga dan baris ketujuh menunjukkan kegagalan proses. Keberhasilan proses diperoleh dari kode sumber yang benar. Dengan menggunakan kode sumber yang benar, sistem telah berhasil melakukan proses *build and test*, *release* serta *deploy* secara otomatis melalui Travis CI. Hasil kompilasi biner *firmware* akan disimpan secara otomatis pada *repository* GitHub

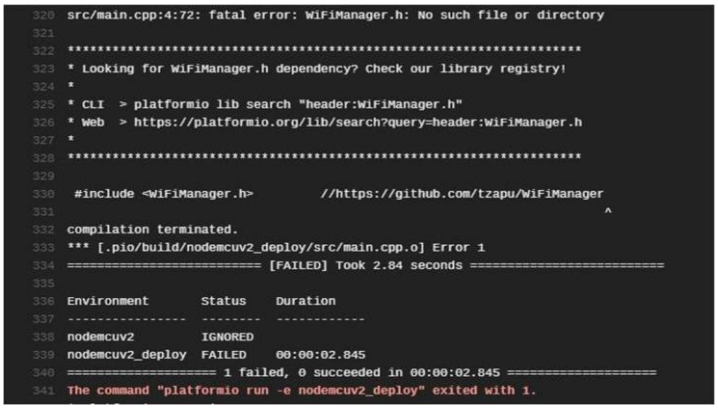
Tabel 2. Hasil *Automatic Test*

No	*Commits Number*	*Build &Test*	*Release*	*Deploy*
1	7560887	*Auto*	*Auto*	*Auto*
2	1079c36	*Auto*	*Auto*	*Auto*

3	181842c	*Failed*	*Failed*	*Failed*
4	77998ac	*Auto*	*Auto*	*Auto*
5	0809c7a	*Auto*	*Auto*	*Auto*
6	a8d19d6	*Auto*	*Auto*	*Auto*
7	bc900c7	*Failed*	*Failed*	*Failed*



Gambar 3. Kegagalan yang Disebabkan karena Penggunaan Variabel yang Belum Dideklarasikan.



Gambar 4. Kegagalan yang Disebabkan karena Kesalahan Penulisan *Library* yang Digunakan.

Kegagalan proses pada baris ketiga dan ketujuh diperoleh dari simulasi kesalahan penulisan kode sumber. Kegagalan pada baris ketiga disebabkan penggunaan variabel yang belum dideklarasikan sebagaimana terlihat pada Gambar 3. Sedangkan kesalahan pada baris ketujuh disebabkan karena kesalahan dalam penulisan *library* yang mengakibatkan sistem tidak dapat menemukan *library* yang digunakan. Simulasi kegagalan ini ditunjukkan pada Gambar 4. Berdasarkan laporan di atas, kesalahan terjadi karena tidak telitinya pengembang dalam menuliskan kode program.

Tabel 3. Hasil *Lead Time Test*

Tahapan Proses	***Lead Time (s)***				
	Commit-7560887	***Commit-1079c36***	***Commit-a8d19d6***	***Commit-77998ac***	***Commit-0809c7a***
Build &	71.14	69.82	***71.53***	69.64	***69.53***

Test					
Release	***5.86***	***5.18***	5.47	5.36	5.47
Deploy	1.12	1.56	***1.86***	***1.09***	1.42
Total (s)	78.12	76.56	***78.86***	***76.09***	76.42

Lead Time Test

Dari ketujuh pengujian yang dilakukan dengan data yang terlihat pada Tabel 2, lima pengujian menunjukkan keberhasilan *automatic test*. Kelima pengujian yang berhasil tersebut diukur waktu yang dibutuhkan oleh setiap tahapan proses untuk melakukan *lead time test*. Pengujian ini hanya dapat dilakukan pada proses yang berhasil dari Tabel 2, karena proses yang gagal tidak akan diproses lebih lanjut oleh sistem. Waktu masing-masing tahapan tersebut dapat dilihat pada Tabel 3. Terlihat untuk uji coba yang dilakukan total waktu yang dibutuhkan berkisar antara 76,09 detik hingga 78,86 detik dengan rerata waktu total 77,21 detik. Secara keseluruhan waktu terbanyak digunakan untuk melakukan proses *build and test* (berkisar antara 69,53 detik hingga 71,53 detik). Lama waktu keseluruhan tahapan tersebut tentunya bergantung pada ukuran kode sumber yang digunakan serta ketersediaan sumber daya peladen pada Travis CI. Waktu yang dibutuhkan untuk *deployment* ke perangkat keras berkisar antara 1,09 detik hingga 1,86 detik dengan rerata waktu *deployment* sebesar 1,41 detik. Waktu *deployment* tersebut juga dipengaruhi oleh kondisi jaringan

SIMPULAN

Dari hasil uji coba terlihat bahwa infrastruktur DevOps yang diusulkan untuk melakukan pengembangan sistem tertanam berbasis ESP8266 telah dapat mendeteksi kesalahan dan berhasil melakukan proses hingga deployment ke perangkat. Lead time test menunjukkan waktu terpanjang terjadi pada proses build and test (rata-rata 77,21 detik). Sedangkan rerata waktu yang dibutuhkan untuk proses deployment ke perangkat sebesar 1,41 detik. Hal ini menunjukkan bahwa proses upgrade firmware ke perangkat cukup singkat. Hasil ini menunjukkan bahwa penerapan DevOps pada sistem tertanam berbasis microcontroller terutama ESP8266 dapat dilakukan. Dengan menggunakan infrastruktur DevOps yang diusulkan, pengembangan sistem tertanam dapat dilakukan secara cepat. Perubahan user requirement dapat dengan cepat ditanggapi oleh pengembang serta dapat segera diterapkan pada sistem tertanam. Penggunaan ESP8266 pada infrastruktur yang diusulkan cukup cocok digunakan untuk perangkat-perangkat IoT yang terhubung dengan jaringan melalui wifi dengan beban komputasi yang tidak terlalu besar, seperti saklar, lampu, power meter maupun pemantauan parameter lingkungan (seperti suhu dan kelembaban). Penggunaan pada microcontroller jenis lain masih perlu diuji lebih lanjut.

REFERENSI

Ahdan, S., Susanto, E. R., & Syambas, N. R. (2019). Proposed Design and Modeling of Smart Energy Dashboard System by Implementing IoT (Internet of Things) Based on Mobile Devices. *2019 IEEE 13th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, 194–199.

Ahmad, I., Samsugi, S., & Irawan, Y. (2022). Penerapan Augmented Reality Pada Anatomi Tubuh Manusia Untuk Mendukung Pembelajaran Titik Titik Bekam

Pengobatan Alternatif. *Jurnal Teknoinfo*, *16*(1), 46. https://doi.org/10.33365/jti.v16i1.1521

Amarudin, A., Saputra, D. A., & Rubiyah, R. (2020). Rancang Bangun Alat Pemberi Pakan Ikan Menggunakan Mikrokontroler. *Jurnal Ilmiah Mahasiswa Kendali Dan Listrik*, *1*(1), 7–13.

Anantama, A., Apriyantina, A., Samsugi, S., & Rossi, F. (2020). Alat Pantau Jumlah Pemakaian Daya Listrik Pada Alat Elektronik Berbasis Arduino UNO. *Jurnal Teknologi Dan Sistem Tertanam*, *1*(1), 29–34.

Borman, R. I., Syahputra, K., Jupriyadi, J., & Prasetyawan, P. (2018). Implementasi Internet Of Things pada Aplikasi Monitoring Kereta Api dengan Geolocation Information System. *Seminar Nasional Teknik Elektro*, *2018*, 322–327.

Budioko, T. (2016). Sistem monitoring suhu jarak jauh berbasis internet of things menggunakan protokol mqtt. *Seminar Nasional Riset Teknologi Informasi*, *1*(30 July), 353–358.

Dita, P. E. S., Al Fahrezi, A., Prasetyawan, P., & Amarudin, A. (2021). Sistem Keamanan Pintu Menggunakan Sensor Sidik Jari Berbasis Mikrokontroller Arduino UNO R3. *Jurnal Teknik Dan Sistem Komputer*, *2*(1), 121–135.

Hafidhin, M. I., Saputra, A., Ramanto, Y., & Samsugi, S. (2020). Alat Penjemuran Ikan Asin Berbasis Mikrokontroler Arduino UNO. *Jurnal Teknik Dan Sistem Komputer*, *1*(2), 26–33.

Hayatunnufus, H., & Alita, D. (2020). SISTEM CERDAS PEMBERI PAKAN IKAN SECARA OTOMATIS. *Jurnal Teknologi Dan Sistem Tertanam*, *1*(1), 11–16.

Imani, M., & Ghassemian, H. (2019). Electrical Load Forecasting Using Customers Clustering and Smart Meters in Internet of Things. *9th International Symposium on Telecommunication: With Emphasis on Information and Communication Technology, IST 2018*, 113–117. https://doi.org/10.1109/ISTEL.2018.8661071

Isnain, A. R., Sintaro, S., & Ariany, F. (2021). *Penerapan Auto Pump Hand Sanitizer Berbasis Iot*. *2*(2), 63–71.

Jayadi, A., Susanto, T., & Adhinata, F. D. (2021). Sistem Kendali Proporsional pada Robot Penghindar Halangan (Avoider) Pioneer P3-DX. *Majalah Ilmiah Teknologi Elektro*, *20*(1), 47. https://doi.org/10.24843/mite.2021.v20i01.p05

Kholidi dkk. (2015). Rancang Bangun Alat Pemberi Pakan dan Pengatur Suhu Otomatis untuk Ayam Pedaging Berbasis Programmable Logic Controller pada Kandang Tertutup. *Rekayasa Dan Teknologi Elektro Rancang*, 86–95.

Kristiawan, N., Ghafaral, B., Borman, R. I., & Samsugi, S. (2021). Pemberi Pakan dan Minuman Otomatis Pada Ternak Ayam Menggunakan SMS. *Jurnal Teknik Dan Sistem Komputer*, *2*(1), 93–105.

Kurniawan, D. E., Iqbal, M., Friadi, J., Borman, R. I., & Rinaldi, R. (2019). Smart Monitoring Temperature and Humidity of the Room Server Using Raspberry Pi and Whatsapp Notifications. *Journal of Physics: Conference Series*, *1351*(1). https://doi.org/10.1088/1742-6596/1351/1/012006

Kurniawan, F., & Surahman, A. (2021). SISTEM KEAMANAN PADA PERLINTASAN KERETA API MENGUNAKAN SENSOR INFRARED BERBASIS

MIKROKONTROLLER ARDUINO UNO. *Jurnal Teknologi Dan Sistem Tertanam*, *2*(1), 7–12.

Nurdiansyah, M., Sinurat, E. C., Bakri, M., & Ahmad, I. (2020). Sistem Kendali Rotasi Matahari Pada Panel Surya Berbasis Arduino UNO. *Jurnal Teknik Dan Sistem Komputer*, *1*(2), 7–12.

Pindrayana, K., Borman, R. I., Prasetyo, B., & Samsugi, S. (2018). Prototipe Pemandu Parkir Mobil Dengan Output Suara Manusia Mengunakan Mikrokontroler Arduino Uno. *CIRCUIT: Jurnal Ilmiah Pendidikan Teknik Elektro*, *2*(2).

Pratama, M. A., Sidhiq, A. F., Rahmanto, Y., & Surahman, A. (2021). Perancangan Sistem Kendali Alat Elektronik Rumah Tangga. *Jurnal Teknik Dan Sistem Komputer*, *2*(1), 80–92.

Pratama Zanofa, A., & Fahrizal, M. (2021). Penerapan Bluetooth Untuk Gerbang Otomatis. *Portaldata.Org*, *1*(2), 1–10.

Puspaningrum, A. S., Firdaus, F., Ahmad, I., & Anggono, H. (2020). Perancangan Alat Deteksi Kebocoran Gas Pada Perangkat Mobile Android Dengan Sensor Mq-2. *Jurnal Teknologi Dan Sistem Tertanam*, *1*(1), 1–10.

Rahmanto, Y., Burlian, A., & Samsugi, S. (2021). SISTEM KENDALI OTOMATIS PADA AKUAPONIK BERBASIS MIKROKONTROLER ARDUINO UNO R3. *Jurnal Teknologi Dan Sistem Tertanam*, *2*(1), 1–6.

Rahmanto, Y., Rifaini, A., Samsugi, S., & Riskiono, S. D. (2020). Sistem Monitoring pH Air Pada Aquaponik Menggunakan Mikrokontroler Arduino UNO. *Jurnal Teknologi Dan Sistem Tertanam*, *1*(1), 23–28.

Ratnasari, T. D., Samsugi, S., Kom, S., & Eng, M. (n.d.). *SETUP MIKROTIK SEBAGAI GATEWAY SERVER PADA SMK PELITA GEDONGTATAAN*.

Rikendry, & Navigasi, S. (2007). *Sistem kontrol pergerakan robot beroda pemadam api*. *2007*(Snati), 1–4.

Riski, M., Alawiyah, A., Bakri, M., & Putri, N. U. (2021). Alat Penjaga Kestabilan Suhu Pada Tumbuhan Jamur Tiram Putih Menggunakan Arduino UNO R3. *Jurnal Teknik Dan Sistem Komputer*, *2*(1), 67–79.

Rumalutur, S., & Ohoiwutun, J. (2018). Sistem Kendali Otomatis Panel Penerangan Luar Menggunakan Timer Theben Sul 181 H Dan Arduino Uno R3. *Electro Luceat*, *4*(2), 43–51. https://doi.org/10.32531/jelekn.v4i2.143

Samsugi, S. (2017). Internet of Things (iot): Sistem Kendali jarak jauh berbasis Arduino dan Modul wifi Esp8266. *ReTII*.

Samsugi, S, & Burlian, A. (2019). Sistem penjadwalan pompa air otomatis pada aquaponik menggunakan mikrokontrol Arduino UNO R3. *PROSIDING SEMNASTEK 2019*, *1*(1).

Samsugi, S, Neneng, N., & Aditama, B. (2018). *IoT: kendali dan otomatisasi si parmin (studi kasus peternak Desa Galih Lunik Lampung Selatan)*.

Samsugi, S, Neneng, N., & Suprapto, G. N. F. (2021). Otomatisasi Pakan Kucing Berbasis Mikrokontroller Intel Galileo Dengan Interface Android. *J-SAKTI (Jurnal Sains Komputer Dan Informatika)*, *5*(1), 143–152.

Samsugi, S, & Silaban, D. E. (2018a). PROTOTIPE CONTROLLING BOX PEMBERSIH

WORTEL BERBASIS MIKROKONTROLER. *ReTII*.

Samsugi, S, & Silaban, D. E. (2018b). Purwarupa Controlling Box Pembersih Wortel Dengan Mikrokontroler. *Prosiding Nasional Rekayasa Teknologi Industri Dan Informasi*, *13*, 1–7.

Samsugi, S, & Suwantoro, A. (2018). Pemanfaatan Peltier dan Heater Sebagai Alat Pengontrol Suhu Air Pada Bak Penetasan Telur Ikan Gurame. *Conf. Inf. Technol*, 295–299.

Samsugi, Selamet, Ardiansyah, A., & Kastutara, D. (2018). Arduino dan Modul Wifi ESP8266 sebagai Media Kendali Jarak Jauh dengan antarmuka Berbasis Android. *Jurnal Teknoinfo*, *12*(1), 23–27.

Samsugi, Selamet, Mardiyansyah, Z., & Nurkholis, A. (2020). Sistem Pengontrol Irigasi Otomatis Menggunakan Mikrokontroler Arduino UNO. *Jurnal Teknologi Dan Sistem Tertanam*, *1*(1), 17–22.

Samsugi, Selamet, Nurkholis, A., Permatasari, B., Candra, A., & Prasetyo, A. B. (2021). Internet of Things Untuk Peningkatan Pengetahuan Teknologi Bagi Siswa. *Journal of Technology and Social for Community Service (JTSCS)*, *2*(2), 174.

Samsugi, Selamet, & Wajiran, W. (2020). IOT: Emergency Button Sebagai Pengaman Untuk Menghindari Perampasan Sepeda Motor. *Jurnal Teknoinfo*, *14*(2), 99–105.

Samsugi, Selamet, Yusuf, A. I., & Trisnawati, F. (2020). Sistem Pengaman Pintu Otomatis Dengan Mikrokontroler Arduino Dan Module Rf Remote. *Jurnal Ilmiah Mahasiswa Kendali Dan Listrik*, *1*(1), 1–6.

Setiawan, M. B., Susanto, T., & Jayadi, A. (2021). PENERAPAN SISTEM KENDALI PID PESAWAT TERBANG TANPA AWAK UNTUK KESETABILAN ROLL, PITCH DAN YAW PADA FIXED WINGS. *The 1st International Conference on Advanced Information Technology and Communication (IC-AITC)*.

Suaidah, S. (2021). Teknologi Pengendali Perangkat Elektronik Menggunakan Sensor Suara. *Jurnal Teknologi Dan Sistem Tertanam*, *02*(02). https://ejurnal.teknokrat.ac.id/index.php/jtst/article/view/1341

Subandi. (2016). *PEMBASMI HAMA SERANGGA MENGGUNAKAN CAHAYA LAMPU BERTENAGA SOLAR CELL*. *9*(1), 86–92.

Surahman, A., Aditama, B., Bakri, M., & Rasna, R. (2021). Sistem Pakan Ayam Otomatis Berbasis Internet Of Things. *Jurnal Teknologi Dan Sistem Tertanam*, *2*(1), 13–20.

Susanto, E. R. (n.d.). *Sistem Penunjang Keputusan Cerdas Spasial Pengendalian Avian Influenza H5n1 Pada Unggas Peternakan Rakyat Non Komersial: Studi Kasus Provinsi Lampung*. Bogor Agricultral University (IPB).

Wantoro, A., Samsugi, S., & Suharyanto, M. J. (2021). Sistem Monitoring Perawatan dan Perbaikan Fasilitas PT PLN (Studi Kasus : Kota Metro Lampung). *Jurnal TEKNO KOMPAK*, *15*(1), 116–130.

Widodo, T., Irawan, B., Prastowo, A. T., & Surahman, A. (2020). Sistem Sirkulasi Air Pada Teknik Budidaya Bioflok Menggunakan Mikrokontroler Arduino Uno R3. *Jurnal Teknik Dan Sistem Komputer*, *1*(2), 1–6.

Yuliana, Y., Paradise, P., & Kusrini, K. (2021). Sistem Pakar Diagnosa Penyakit Ispa Menggunakan Metode Naive Bayes Classifier Berbasis Web. *CSRID (Computer*

Science Research and Its Development Journal), *10*(3), 127. https://doi.org/10.22303/csrid.10.3.2018.127-138

Yulianti, T., Samsugi, S., Nugroho, P. A., & Anggono, H. (2021). Rancang Bangun Pengusir Hama Babi Menggunakan Arduino dengan Sensor Gerak. *JTST*, *2*(1), 21–27.

Yurnama, T. F., & Azman, N. (2009). Perancangan Software Aplikasi Pervasive Smart Home. *Snati*, *2009*(Snati), E2–E5.

Zanofa, A. P., Arrahman, R., Bakri, M., & Budiman, A. (2020). Pintu Gerbang Otomatis Berbasis Mikrokontroler Arduino UNO R3. *Jurnal Teknik Dan Sistem Komputer*, *1*(1), 22–27.