Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 9, September 2018, hlm. 2806-2813 http://j-ptiik.ub.ac.id

Implementasi Perangkat Rumah Cerdas Pervasif Menggunakan Komunikasi *Websocket* Dan Struktur Data JSON

Ahmad Mustafidul Ibad[1], Sabriansyah Rizqika Akbar[2], Dahnial Syauqy[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]ahmadafid@gmail.com, [2]sabrian@ub.ac.id, [3]dahnial87@ub.ac.id

Abstrak

Perkembangan teknologi telah memicu pergeseran mendasar dalam gaya hidup modern. Salah satu penerapan dalam teknologi yang saat ini banyak dikupas secara mendalam adalah rumah cerdas *(smart home)*. Perangkat dari teknologi rumah cerdas sangat beragam sehingga sulit untuk menyelaraskan kinerja sistem yang dibuat. Hal tersebut diatasi dengan membagi sistem menjadi sub-sistem yang lebih kecil sehingga akan memudahkan dalam manajemen sistem yang dibuat. Hal itu disebut juga sebagai *modular system*. Adanya *self-management devices* sangat penting agar perangkat pada rumah cerdas dapat memanajemen dirinya sendiri. *Pervasive devices* merupakan istilah yang digunakan untuk perangkat yang meresap dalam kehidupan sehari-hari yang memberikan layanan serta kemudahan kepada penggunanya. Untuk mencapai hal itu, *pervasive devices* harus mampu bekerja secara mandiri. Penelitian ini merancang dan menerapkan sistem yang dapat mendeteksi adanya raspberry pi *(pervasive devices)* secara otomatis pada jaringan lokal dan secara mandiri sistem dapat menyiapkan serta menyediakan layanan dimana pengguna dapat menggunakan layanan tersebut yaitu kendali led melalui *web browser*. Teknologi atau metode komunikasi antara *web browser* dan raspberry pi yang digunakan adalah *websocket*, sedangkan antara raspberry pi dengan wemos sebagai mikrokontroler untuk led yaitu menggunakan socket UDP *(User Datagram Protocol)*. Pengujian dilakukan dengan menghitung waktu penemuan *(discovery time)* untuk menemukan raspberry pi pada jaringan lokal serta waktu respon layanan (on off, *dimming* dan rgb). Dari hasil pengujian, waktu penemuan didapatkan rata-rata waktu sebesar 4,23 detik dan rata-rata waktu respon layanan sebesar 0,52 detik.

Kata kunci: *smart home, pervasive device, modular system*

Abstract

Technological development has trigger fundamental shift in modern lifestyle. One of application in technology which is currently deeply peeled is smart home. The device of smart home technology are very diverse so difficult to align the performance of the created system. It is solved by dividing the system into smaller sub-system so it will ease in management system created. It is also called a modular system. The existence of self-management devices is very important so that the device in the smart home can manage itself. Pervasive devices are terms used for devices that are pervasive in everyday life that provide service and convenience to its user. To achieve this, pervasive devices must be able to work independently. This research design and implement system that can detect the presence of raspberry pi (pervasive devices) automatically on the local network and independently the system can prepare and provide service where the user can use the service that is control led via web browser. The technology or method of communication between web browser and raspberry pi used is websocket, whereas between raspberry pi with wemos as microcontroller for led is using UDP (User Datagram Protocol) socket. Testing is done by calculating discovery time to find raspberry pi on local network and service response time (on off, dimming and rgb). From the test results, the time of the discovery got an average time amount 4,23 seconds and the average service response time amount 0,52 seconds.

Keywords: *smart home, pervasive device, modular system*

1. PENDAHULUAN

Perkembangan teknologi telah memicu pergeseran mendasar dalam gaya hidup modern (Tiwari, Sewaiwar & Chung, 2015). Teknologi

semakin beragam, penerapan teknologi di dalam rumah cerdas adalah salah satu pemanfaatan teknologi dimana perangkat bisa saling terhubung untuk memberikan layanan kepada pemilik rumah. Pemilik atau pengguna dari teknologi tersebut menginginkan kemudahan dalam pengoperasiannya tak terkecuali dalam hal manajemen perangkat. Adanya *self-management devices* yang artinya perangkat dapat memanajemen dirinya sendiri dan otomatis menyediakan layanan kepada pengguna adalah hal yang perlu ada di dalam rumah cerdas.

Perangkat di dalam rumah cerdas saling berkoordinasi untuk menciptakan hunian yang nyaman. Mereka berkoordinasi satu sama lain menggunakan teknologi komunikasi. *Websocket* adalah satu dari banyak teknologi komunikasi yang lainnya. *Websocket* memungkinkan aliran data *full duplex* dan komunikasi dua arah secara bersamaan antara klien dan server, cocok digunakan untuk aplikasi web secara *real-time* (Wang, Salim & Moskovits, 2013).

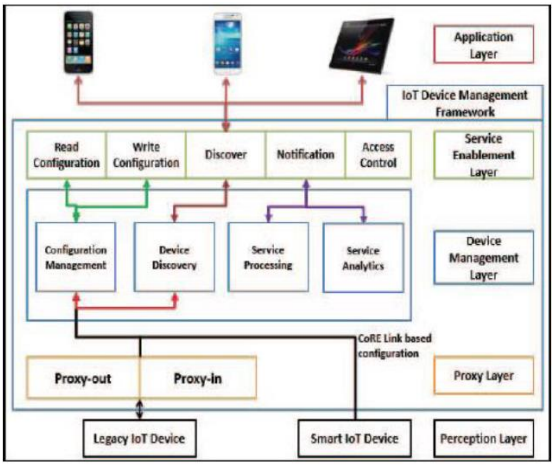
Integrasi rumah cerdas dengan *websocket* khususnya untuk perangkat yang membutuhkan aliran data secara *real-time* akan meningkatkan keakuratan waktu komunikasi agar tercipta layanan dengan performa yang baik. Perangkat rumah cerdas harus mampu menyediakan layanan jangka panjang untuk pemakainya. Teknologi *websocket* mendukung hal tersebut yakni dengan menggunakan fitur *long polling*.

Fitur *long polling* pada *websocket* akan menahan permintaan HTTP *(HTTP request)* yang terbuka guna memberikan jangka waktu sesi koneksi yang lama (Muller, 2014). Dengan karakteristiknya itu, maka *websocket* efisien digunakan bersamaan dengan tipe data dan struktur data apapun. Javascript Object Notation (JSON) adalah salah satu struktur data yang sering digunakan dalam aplikasi saat ini. JSON memberikan kinerja sistem yang lebih baik dan didukung penuh serta cocok untuk aplikasi Javascript sehingga memberikan kinerja yang signifikan daripada XML, dimana XML masih membutuhkan *library* tambahan untuk mengambil data dari obyek DOM (Nuseitov, et al., 2015).

2. TINJAUAN PUSTAKA

Sebuah jurnal internasional yang berjudul *"IoT Device Management Framework for Smart Home Scenarios"* oleh Thinagaran Perumal tahun 2015, menyebutkan bahwa paradigma *Internet of Things* membutuhkan konektivitas pervasif untuk miliaran perangkat heterogen. Semakin hari, pertumbuhan perangkat IoT di lingkungan rumah cerdas semakin meningkat untuk dapat menyediakan layanan dan aplikasi baru. Oleh karena itu, manajemen perangkat tersebut menjadi sangat sulit karena masalah kompleksitas. Disebutkan bahwa masalah dalam pengelolaan perangkat IoT adalah heterogenitas.

Berdasarkan Gambar 1, langkah yang diusulkan untuk mengatasi masalah heterogenitas adalah menggunakan *framework* yang terdiri dari beberapa lapisan yaitu *proxy layer, device management layer: "service processing, configuration management, service analytics, device discovery", service enamblement layer*.



Gambar 1. Framework Manajemen IoT untuk Rumah Cerdas

Sumber: (Perumal, T., Datta, S.K. & Bonnet, C. 2015)

Tabel 1. Lapisan pada *framework* manajemen IoT untuk rumah cerdas

Sumber: (Perumal, T., Datta, S.K. & Bonnet, C. 2015)

No.	Lapisan *(Layer)*	Modul
1.	*Proxy Layer*	-
2.	*Device Management Layer*	*Service processing, configuration management, service analytics, device discovery*
3.	*Service Enablement Layer*	*Read configuration, write configuration*

		discover, notification, access control
4.	*Application Layer*	-

- *Proxy Layer:*
  Lapisan ini mencakup beberapa *driver* untuk protokol dan teknologi komunikasi. Menurut jurnal tersebut, hal itu perlu karena perangkat IoT dapat berkomunikasi menggunakan BLE, NFC, ZigBee dan Wifi. *Proxy layer* membuat *framework* dapat berinteraksi dan melakukan manajemen perangkat. Dalam lapisan ini terdapat *proxy in* dan *proxy out* yang memungkinkan pembuatan konfigurasi berbasis CORE *link format (Core link based configuration).*
- *Device Management Layer:*
  Lapisan ini menyediakan beberapa fungsi seperti *configuration management, device discovery, service processing* dan *service analytics.* Fungsi tersebut dikembangkan dan dipaparkan melalui *web service* pada *service enablement layer*.
- *Service Enablemenet Layer:*
  Fungsionalitas inti yang disebutkan di atas dipaparkan ke konsumen menggunakan RESTful *web service.* Hal ini semakin meningkatkan interoperabilitas dengan berbagai layanan IoT untuk rumah cerdas. Lapisan ini juga menerapkan kebijakan *access control* yang membatasi akses pengguna ke perangkat yang terotorisasi saja. *Smartphone* konsumen dapat terhubung ke lapisan ini untuk mengakses berbagai fungsi *framework* perangkat IoT.

Sebuah sumber jurnal lain yang berjudul *"Design and Implementation of Modular Home Automation based on Wireless Network, REST API, and Websocket"* oleh Atas Hasibuan tahun 2015, menyebutkan bahwa rata-rata sistem rumah cerdas terkendala pada programibiltas dan kustomisasi. Sistem modular adalah hal yang biasa digunakan untuk mengubah desain kompleks menjadi modul yang lebih kecil. Dengan sistem modular, pengguna dapat memodifikasi sistem dengan desainnya sendiri. Dalam jurnal tersebut, dipaparkan bahwa sistem yang dibangun memiliki tiga fitur utama yaitu *electronic device status, electronic device control* dan *electronic device automation*. Perangkat keras yang digunakan adalah router, kontroler sistem, modul komunikasi nirkabel, mikrokontroler untuk menerima perintah dari kontroler sistem.

Latar belakang penelitian tersebut adalah meningkatnya kreativitas seseorang yang ingin membangun sesuatu berdasarkan ide, gagasan dan kebutuhan mereka. Setiap orang memilikinya berbeda-beda. Oleh karena itu, masing-masing membutuhkan desain otomasi rumah cerdas yang berbeda. Tetapi, banyak produk otomasi rumah cerdas telah menyedikan *general template* dimana semua orang akan mendapatkan fitur yang sama. Beberapa fitur mungkin tidak cocok dengan sebagian orang, karena kebutuhannya yang berbeda. Oleh karena itu, seperti yang dijelaskan oleh Atas Hasibuan, et al., (2015), solusi yang ditawarkan adalah menggunakan *modular home automation system* dimana orang dapat menyesuaikan desain rumah cerdas mereka sesuai dengan ide, gagasan dan kebutuhannya. Dengan solusi tersebut, otomasi rumah cerdas akan mudah dipasang dan dimodifikasi. Dengan sistem modular, perangkat satu sama lain memiliki sifat *independence*.

3. METODOLOGI PENELITIAN

Penelitian ini diawali dengan melakukan studi literatur dari beberapa sumber yaitu jurnal, *e-book*, makalah dan dasar teori mengenai penelitian yang dilakukan. Kemudian dilanjutkan dengan menganalisis kebutuhan sistem yang terdiri dari kebutuhan perangkat keras dan perangkat lunak. Perancangan sistem dilakukan setelah menganalisis keseluruhan sistem. Untuk dapat memahami perancangan sistem secara keseluruhan, maka hal ini akan digambarkan dengan diagram blok. Implementasi sistem akan dilakukan sesuai dengan perancangan sistem yang telah dibuat sebelumnya. Setelah perancangan dibuat maka selanjutnya bisa melakukan implementasi sistem yang kemudian dilakukan pengujian sistem yang telah dibuat. Kesimpulan diambil setelah melakukan pengujian dan analisis hasil pengujian sistem.

3.1. Analisis Kebutuhan Sistem

Analisis kebutuhan sistem bertujuan untuk mendapatkan semua kebutuhan yang dibutuhkan oleh sistem. Analisis kebutuhan dilakukan dengan mengidentifikasi kebutuhan sistem yang terdiri dari kebutuhan perangkat keras dan perangkat lunak. Dengan adanya pengidentifikasian ini nantinya akan

mempermudah dalam melakukan desain dan pembuatan sistem.

3.1.1. Kebutuhan Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah sebagai berikut:

- Raspberry Pi
- LED dan RGB LED
- Wemos D1 R2 *(based on ESP8266)*

3.1.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut:

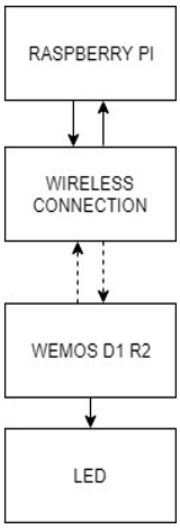
- *Integrated Development Environment* (IDE) Arduino
- Raspbian
- Python
- Modul *websocket* untuk Python
- Modul tornado
- Perangkat lunak basis data SQLITE3
- Pustaka ArduinoJson
- Pustaka LED
- Pustaka FiniteStateMachine
- Pustaka ESP8266

3.2. Perancangan Sistem

Perancangan sistem dilakukan setelah menganalisis keseluruhan sistem. Perancangan sistem akan digambarkan dalam bentuk diagram blok seperti pada gambar 2.

LED merupakan obyek dari penelitian yang terdiri dari LED biasa dan RGB LED. LED terhubung ke wemos. Wemos inilah yang menghubungkan LED sebagai obyek dengan raspberry pi sebagai server program kendali LED tersebut. Terdapat tanda dengan garis putus-putus yang berarti hal tersebut merupakan koneksi wifi.

Wemos dan raspberry pi terhubung ke akses poin dalam jaringan lokal dimana keduanya dapat saling berkomunikasi dan raspberry pi dapat mengirim data perintah kendali LED ke wemos, yaitu on off, dimming dan rgb. Perintah kendali LED dimasukkan oleh pengguna melalui tampilan antarmuka web pada web browser yang telah disediakan.



Gambar 2. Blok Diagram Sistem

3.3 Implementasi

Implementasi sistem akan dilakukan sesuai dengan perancangan sistem yang telah dibuat sebelumnya. Pada bagian ini terdapat beberapa proses implementasi, yaitu *package* program pengendalian led, implementasi program *state machine* dan data JSON pada wemos menggunakan media komunikasi wifi.

3.4 Pengujian dan Analisis

Pengujian dilakukan agar mengetahui bahwa sistem telah bekerja dengan baik sesuai dengan latar belakang dan spesifikasi yang melandasinya. Terdapat dua pengujian yang dilakukan yaitu:

1. Pengujian *discovery time* yaitu waktu yang diperlukan wemos untuk menemukan raspberry pi pada jaringan
2. Pengujian *response time* layanan yaitu on off, *dimming* dan RGB LED.

4. HASIL DAN PEMBAHASAN

Untuk mengetahui hasil dari sistem yang dibuat dilakukan pengujian seperti berikut.

4.1 Pengujian *Discovery Time*

Discovery time adalah waktu yang diperlukan wemos untuk menemukan raspberry dan terhubung dengan raspberry. Tujuan dari pengujian *discovery time* adalah untuk melihat estimasi waktu yang diperlukan wemos untuk dapat menemukan raspberry pi apakah didapatkan waktu yang semestinya dan tidak terlalu lama untuk dapat menemukan raspberry di dalam jaringan lokal.

Tabel 2. Pengujian *Discovery Time*

Percobaan ke-	Waktu *(discovery time)*
1	3,65 s
2	3,55 s
3	4,79 s
4	4,03 s
5	4,77 s
6	3,99 s
7	3,54 s
8	3,84 s
9	4,02 s
10	3,45 s
11	4,08 s
12	3,61 s
13	4,00 s
14	3,55 s
15	3,65 s
16	4,03 s
17	4,08 s
18	4,01 s
19	3,54 s
20	7,37 s
21	6,03 s
22	3,57 s
23	4,00 s
24	6,10 s
25	4,71 s
rata-rata	4,23 s

Berdasarkan pengujian *discovery time* yang telah dilakukan, didapatkan hasil estimasi waktu yang bervariatif dimana waktu tersebut diperoleh dengan mengurangkan *end_time* sebagai waktu akhir yang telah didefinisikan dengan *start_time* (waktu awal). Waktu dihitung saat raspberry siap mendengarkan koneksi dari wemos hingga wemos dapat menemukan raspberry pi dan data *service and device description* dikirimkan ke raspberry pi dan ditampilkan pada *web browser*. Didapatkan rata-rata estimasi waktu penemuan sebesar 4,23 detik. Wemos dapat bekerja dengan baik untuk menemukan raspberry pi pada jaringan.

4.2 Pengujian On Off LED

Pengujian on off dilakukan untuk mengetahui waktu respon (*response time*) pada layanan on off LED pada wemos.

Tabel 3. Pengujian On Off LED

Percobaan ke-	Kondisi	Waktu
1	On	0,52 s
2	Off	0,52 s
3	On	0,70 s
4	Off	0,52 s
5	On	0,52 s
6	Off	0,52 s
7	On	0,52 s
8	Off	0,52 s
9	On	0,53 s
10	Off	0,53 s
11	On	0,52 s
12	Off	0,52 s
13	On	0,52 s
14	Off	0,53 s
15	On	0,52 s
16	Off	0,52 s
17	On	0,51 s
18	Off	0,52 s
19	On	0,52 s
20	Off	0,52 s
21	On	0,52 s
22	Off	0,52 s
23	On	0,51 s
24	Off	0,53 s
25	On	0,54 s
rata-rata	0,52 s	

Berdasarkan pengujian waktu respon layanan on off yang telah dilakukan, didapatkan hasil waktu yang bervariatif. Waktu tersebut terhitung sejak pengguna mengirimkan nilai perintah melalui *web browser* ke wemos. *Start_time* terhitung ketika raspberry pi mengirimkan perintah ke wemos sedangkan *end_time* terhitung saat wemos memberikan respon dan data yang berisi nilai perintah pada basis data telah di-*update*. Pengiriman perintah nilai on off dilakukan pada *service_id* 00011. Pengujian dapat berjalan dengan baik dengan rata-rata hasil waktu respon on off adalah 0,52 detik.

4.3 Pengujian *Dimming* LED

Pengujian ini dilakukan untuk mengetahui waktu respon (*response time*) layanan *dimming* LED pada wemos.

Tabel 4. Pengujian *Dimming* LED

No	Nilai PWM	Waktu
1	5	0,52 s
2	15	0,57 s
3	25	0,52 s
4	35	0,53 s
5	45	0,52 s
6	55	0,51 s
7	65	0,53 s
8	75	0,52 s
9	85	0,53 s
10	95	0,52 s
11	105	0,52 s
12	115	0,53 s
13	125	0,53 s
14	135	0,52 s
15	145	0,52 s
16	155	0,53 s
17	165	0,53 s
18	175	0,52 s
19	185	0,52 s
20	195	0,52 s
21	205	0,52 s
22	215	0,52 s
23	225	0,54 s
24	235	0,53 s
25	245	0,53 s
rata-rata	0,52 s	

Berdasarkan pengujian *dimming* LED yang telah dilakukan, didapatkan data estimasi waktu respon layanan *dimming* yang bervariatif. *Start_time* terhitung ketika raspberry pi mengirimkan perintah ke wemos sedangkan *end_time* terhitung saat wemos memberikan respon dan data yang berisi nilai perintah pada basis data telah di-*update*. Pengiriman perintah nilai *dimming* dilakukan pada *service_id* 00012. Pengujian dapat berjalan dengan baik dengan rata-rata hasil waktu respon *dimming* adalah 0,52 detik.

4.4 Pengujian RGB LED

Pengujian ini dilakukan untuk mengetahui waktu respon (*response time*) layanan *red green blue* (RGB) pada wemos.

Tabel 5. Pengujian RGB LED

No	Nilai RGB (Warna)	Waktu
1	255_0_0 (*red*)	0,52 s
2	0_255_255 (*aqua*)	0,52 s
3	0_255_0 (*lime*)	0,53 s
4	255_255_255 (*white*)	0,52 s
5	128_128_128 (*gray*)	0,52 s
6	0_255_127 (*springGreen*)	0,53 s
7	46_139_87 (*seaGreen*)	0,52 s
8	154_205_50 (*yellowGreen*)	0,52 s
9	107_142_35 (*oliveDrab*)	0,52 s
10	128_128_0 (*olive*)	0,52 s
11	102_205_170 (*mediumAquamarine*)	0,53 s
12	143_188_143 (*darkSeaGreen*)	0,52 s
13	32_178_170 (*lightSeaGreen*)	0,52 s
14	0_139_139 (*darkCyan*)	0,52 s
15	255_0_255 (*magenta*)	0,52 s
16	186_85_211 (*mediumOrchid)*	0,52 s
17	216_191_216 (*thistle*)	0,51 s
18	255_255_0 (*yellow*)	0,52 s
19	255_215_0 (*gold*)	0,52 s
20	255_165_0 (*orange*)	0,52 s

21	255_99_71 (*tomato*)	0,52 s
22	255_69_0 (*orangeRed*)	0,52 s
23	135_206_235 (*skyBlue*)	0,52 s
24	64_224_208 (*turquoise*)	0,52 s
25	25_25_112 (*midnightBlue*)	0,52 s
rata-rata	0,52 s	

Berdasarkan pengujian RGB LED yang telah dilakukan, didapatkan data estimasi waktu respon layanan *rgb* yang bervariatif. *Start_time* terhitung ketika raspberry pi mengirimkan perintah ke wemos sedangkan *end_time* terhitung saat wemos memberikan respon dan data yang berisi nilai perintah pada basis data telah di-*update*. Pengiriman perintah nilai rgb dilakukan pada *service_id* 00013. Pengujian dapat berjalan dengan baik dengan rata-rata hasil waktu respon *dimming* adalah 0,52 detik.

5. KESIMPULAN DAN SARAN

Kesimpulan yang dapat diambil dari penelitian ini dan pengujian yang telah dilakukan adalah waktu *discovery time* dan waktu respon layanan diperoleh dengan mengurangkan waktu akhir *(end time)* dan waktu awal *(start time).* Pada *discovery time*, *start time* didefinisikan pada saat *socket* Python pertama kali dijalankan dan *end time* didefinisikan setelah data *device* dan *service description* tampil pada *web browser*

Waktu respon time on off, *dimming* dan rgb diperoleh dari *end time* yang didefinisikan saat sistem memperoleh respon dari *action* kendali led yang dilakukan dan sistem dapat menampilkan hasil respon ke pengguna melalui *web browser*. *End time* dikurangi dengan *start time* yang diperoleh pada awal perintah kendali led dikirim ke wemos. Pengujian on off, *dimming* dan rgb diperoleh hasil pengujian yang dapat berjalan dengan baik dan memerlukan waktu rata-rata respon sebesar 0,52 detik.

Saran untuk penelitian selanjutnya dapat menerapkan metode komunikasi *websocket* untuk koneksi antar raspberry pi atau dengan wemos ataupun mikrokontroler yang lain. Dapat mengembangkan tampilan pada antarmuka *web browser* dan melakukan integrasi data JSON ke dalam lingkup yang lebih luas. Membuat wemos dan raspberry pi dapat berkomunikasi melalui internet global sehingga meningkatkan fleksibilitas dan kegunaan sistem dan dapat menambah obyek yang digunakan pada penelitian selanjutnya serta melakukan standarisasi *discovery time* dan waktu respon layanan agar diperoleh standar waktu respon yang lebih sesuai.

6. DAFTAR PUSTAKA

Tiwari, S.V., Sewaiwar, A. & Chung, Y.H., 2015. *Smart Home Technologies using Visible Light Communication*, [e-journal]. Tersedia melalui: IEEE Xplore Digital Library <ieeexplore.ieee.org/document/7066453> [diakses 21 Agustus 2017]

Suresh, S. & Sruthi P.V., 2015. *A Review on Smart Home Technology*, [e-journal]. Tersedia melalui: IEEE Xplore Digital Library <ieeexplore.ieee.org/document/7453832> [diakses 21 Agustus 2017]

Lombardi, A., 2015. *Websocket: Light-weight Client-Server Communications*. Sebastopol: O'Reilly Media, Inc.

Wang, V., Salim, F. & Moskovits, P., 2013. *The Definitive Guide to HTML5 Websocket*. New York: SSBM Finance Inc.

Muller, G.L., 2014. *HTML5 Websocket protocol and its application to distributed computing*. Bedfordshire: Cranfield University.

Nurseitov, N. et al., 2015. *Comparison of JSON and XML Data Interchange Formats: A Case Study*, [e-journal]. Tersedia melalui: Montana State University Library <https://www.cs.montana.edu/izurieta/pubs/caine2009.pdf> [Diakses 22 Agustus 2017]

Sunehra, D. & Goud, V.S., 2016. *Attendance Recording and Consolidation System using Arduino and Raspberry Pi*, [e-journal]. Tersedia melalui: IEEE Xplore Digital Library <ieeexplore.ieee.org/document/7955639> [diakses 25 Agustus 2017]

Akbar, S.R., et al., 2015. *Implementasi Purwarupa Perangkat Rumah Cerdas Pervasif Berbasis Protokol Universal Plug and Play (UPnP) dan Raspberry Pi General Purpose Input/Output (GPIO)*, [e-journal]

2(2), 116-124. Tersedia melalui: Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) <jtiik.ub.ac.id/index.php/jtiik/article/view/143/pdf> [diakses 25 Agustus 2017]

Kuhlman, D, 2013. *A Python Book: Beginning Python, Advanced Python, and Python Exercises*. [pdf] Dave Kuhlman. Tersedia di: <www.davekuhlman.org/python_book_01.pdf> [diakses 25 Agustus 2017]

Tutorials Point, 2016. *Websocket: tutorialspoint simply easy learning* www.tutorialspoint.com. [pdf] Tutorials Point. Tersedia di: <https://www.tutorialspoint.com/websockets/websockets_tutorial.pdf> [diakses 25 Agustus 2017]

Official Raspberry Pi Magazine, 2016. *The Official Raspbery Pi Projects Book: from the makers of magpi*. [pdf] Official Raspberry Pi Magazine. Tersedia di: <https://raspberrypi.org/magpi-issues/MagPi49.pdf> [diakses 26 Agustus 2017]

Tutorials Point, 2016. *Javascript Object Notation: tutorialspoint simply easy learning* www.tutorialspoint.com. [pdf] Tutorials Point. Tersedia di: <https://www.tutorialspoint.com/json/json_tutorial.pdf> [diakses 26 Agustus 2017]

Goodrich, M.T, et al., 2013. *Data Structures and Algorithms in Python*. Hoboken: John Wiley & Sons, Inc.

Tutorials Point, 2017. *SQLite sql database engine: tutorialspoint simply easy learning* www.tutorialspoint.com. [pdf] Tutorials Point. Tersedia di: <https://www.tutorialspoint.com/sqlite/sqlite_tutorial.pdf> [diakses 07 September 2017]

Allen, G. & Owens, M., 2010. *The Definitive Guide to SQLite*. New York: Springer Science+Business Media, LLC.

Opdenacker, M., 2017. *Hotplugging with udev: embedded linux* http://free-electrons.com. [pdf] Free-electrons. Tersedia di: <https://free-electrons.com/doc/legacy/udev/udev.pdf> [diakses 08 September 2017]

Szczys, M., 2009. *Writing Udev Rules*. [pdf] Mike Szczys. Tersedia di: <https://wiki.robojackets.org/images/5/5b/How_to_write_udev_rules_-_Hack_a_Day.pdf> [diakses 08 September 2017]

The Tornado Authors, 2017. *Tornado Documentation Release 5.0.dev1*. [pdf] The Tornado Authors. Tersedia di: <https://media.readthedocs.org/pdf/tornado/latest/tornado.pdf> [diakses 09 September 2017]

Dory, M., Parrish, A. & Berg, B., 2012. *Introduction to Tornado: Modern Web Applications with Python*. Sebastopol: O'Reilly Media, Inc.

Guardia, Cdl., 2016. *Python Web Frameworks.* Sebastopol: O'Reilly Media, Inc.

Tutorials Point, 2015. *JavaScript Language: tutorialspoint simply easy learning* www.tutorialspoint.com. [pdf] Tutorials Point. Tersedia di: <https://www.tutorialspoint.com/javascript/javascript_tutorial.pdf> [diakses 10 September 2017]

Tutorials Point, 2015. *JQUERY web application library: tutorialspoint simply easy learning* www.tutorialspoint.com. [pdf] Tutorials Point. Tersedia di: <https://www.tutorialspoint.com/jquery/jquery_tutorial.pdf> [diakses 10 September 2017]

Bhasker, L.T., 2013. International Journal Of Engineering And Computer Science. *Pervasive Computing Issues, Challenges and Applications*, [e-journal] 2(12), 3337-3339. Tersedia melalui: <https://www.ijecs.in/issue/v2-i12/1%20ijecs.pdf> [diakses 12 September 2017]

Goyal, M.J. & Dadhich, M.A., 2015. International Journal of Advanced Research in Computer and Communication Engineering. *Pervasive Computing*, [e-journal] 4(12). Tersedia melalui: <https://www.ijarcce.com/upload/2015/december-15/IJARCCE%203.pdf> [diakses 12 September 2017]