BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi landasan kepustakaan yang berisi kajian pustaka dan dasar teori dalam menunjang proses penelitian. Kajian pustakan ini membahas terkait penelitian-penelitian yang sudah ada dan mendukung dalam proses penelitian yang diusulkan. Dasar teori ini berisi berbagai teori yang diperlukan dalam penelitian ini.

2.1 Kajian Pustaka

Kajian pustaka membahas berbagai penelitian yang telah dilakukan sebelumnya sehingga menjadi pendukung dalam proses penelitian yang diusulkan. Pada penelitian ini, kajian pustaka diambil dari beberapa penelitian yang sudah ada. Penelitian yang dilakukan oleh Akbar, dkk (2016) yang berjudul "*Pervasive Device and Service Discovery Protocol In XBee Sensor Network"* yakni mengaplikasikan suatu sistem yang berfungsi untuk mengidentifikasi suatu layanan dan perangkat yang berada di jaringan sensor yang menggunakan protokol XBee. Penulis menerapkan protokol pervasif pada mikrokontroler arduino sebagai *node* klien yang diintegrasikan dengan XBee *transceiver* dan raspberry pi sebagai *gateway* yang juga diintegrasikan dengan XBee *transceiver*. Berdasarkan hasil percobaan, *gateway* tersebut mampu menemukan suatu perangkat dan layanan baru didalam jaringan sensor dalam waktu rata-rata 4,13 detik serta waktu *round trip* rata-rata untuk *request* dan *response* data dari *gateway* yakni sebesar 0.201 detik.

Penelitian lainnya yang dilakukan oleh Suprayogi, dkk (2016) yang berjudul "Implementasi *Pervasive Service Discovery Protocol* pada Rumah Cerdas Berbasis NRF24L01" yakni mengusulkan sistem yang sama dengan penelitian sebelumnya yakni sistem yang mampu mengidentifikasi suatu layanan dan perangkat baru pada suatu jaringan sensor yang berbasis *wireless radio* menggunakan NRF24L01. Penulis menerapkan protokol pervasif pada arduino sebagai *node* klien yang diintegrasikan dengan NRF24L01 sebagai modul komunikasinya serta memiliki sensor suhu dan lampu serta memiliki *gateway* yang menggunakan arduino yang diintegrasikan dengan NRF24L01. Berdasarkan hasil percobaan, waktu yang dibutuhkan 1 *node* klien untuk terhubung dengan *gateway* yakni 1011 milidetik dan waktu yang dibutuhkan 2 *node* klien untuk terhubung dengan *gateway* secara bersamaan masing-masing memiliki waktu rata-rata 1011 milidetik dan 1123,2 milidetik.

Penelitian lainnya yang dilakukan oleh Rochman, dkk (2016) yang berjudul "Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol MQTT pada *Smarthome*" menyatakan penggunaan MQTT sebagai suatu protokol yang memiliki integritas datanya mencapai 100% untuk mengendalikan 2 buah lampu led dengan mengumpulkan data sensor suhu dan intensitas cahaya. Penulis menggunakan mikrokontroler Wemos D1 R2 untuk *node* sensor dan *node* aktuator yang kemudian mengirimkan data sensor tersebut menggunakan *Wi-Fi*

yang akan ditangkap datanya oleh komputer dan di dalam komputer tersebut terdapat aplikasi yang akan membaca data dari sensor.

Berdasarkan penelitian-penelitian diatas, dapat diketahui bahwa metode pengenalan suatu perangkat dan layanan dari suatu sensor dan aktuator telah berhasil diimplementasikan pada objek rumah cerdas dengan menggunakan berbagai pilihan jaringan sensor. Protokol komunikasi MQTT juga telah diimplementasikan untuk komunikasi antar device pada objek rumah cerdas dengan menghasilkan integritas datanya mencapai 100% . Fokus pada penelitian ini adalah mengimplementasikan sistem pengenalan perangkat dan layanan sensor dan aktuator berbasis arsitektur *publish-subscribe* mengunakan protokol MQTT pada objek rumah cerdas.

2.2 Dasar Teori

Dasar teori membahas berbagai teori dasar untuk menunjang proses penilitian. Pada subbab dasar teori akan dijelaskan referensi dan teori dasar pendukung mengenai gambaran umum tentang rumah cerdas serta salah satu contoh protokol yang menerapkan arsitektur *publish-subscribe*, berbagai protokol atau mekanisme yang digunakan untuk mengidentifikasi suatu perangkat dan layanan dan komponen penunjang lainnya seperti mikrokontroler serta format teks untuk pertukaran informasi menggunakan json.

2.2.1 Rumah Cerdas

Rumah cerdas merupakan salah satu contoh yang sangat populer dari sekian banyak aplikasi yang menerapkan teknologi IoT (*Internet of Things*), dimana aplikasi ini memungkinkan untuk meningkatkan gaya hidup manusia. Belakangan ini, manusia dikelilingi oleh ribuan bahkan jutaan lebih perangkat elektronik, sehingga dengan adanya sensor dan aktuator yang terintegrasi dengan perangkat elektronik tersebut maka akan membantu memanfaatkan energi secukupnya dan menambah kenyamanan hidup (Hussain, 2016).

Orang-orang yang terlahir dalam disabilitas, sakit keras dan kekurangan lainnya akan sangat merasa diuntungkan dengan adanya solusi ini, dimana akan menyetarakan kemudahan memudahkan orang tersebut dengan orang-orang yang normal dalam beraktifitas sehari-hari. Salah satu contohnya, yakni penerapan lampu otomatis, kontrol suhu, dan aplikasi lainnya yang turut membantu orang-orang yang memiliki banyak kekurangan sehingga terdapat kendala untuk menjalani aktifitasnya (Prasetio, 2017).

2.2.2 Sistem Pervasif

Sistem pervasif pada teknologi *embedded* maupun *Internet of Things* memungkin untuk melakukan suatu komputasi yang berfungsi untuk menemukan suatu layanan yang diinginkan berdasarkan berbagai konteks (preferensi pengguna, lokasi pengguna, aktifitas pengguna, dan lain lain) serta berinteraksi dengan pengguna serta lingkungan untuk membantu pengguna dalam kehidupan sehari-hari tanpa ada gangguan dari pengguna (Bhatti, 2014)

Salah satu tujuan model sistem ini dikembangkan untuk meningkat pengalaman pengguna dalam menggunakan suatu perangkat dengan tidak adanya gangguan dari pengguna sehingga perangkat tersebut seakan-akan tidak dapat namun tetap dapat menjalankan fungsionalitasnya. Penemuan suatu layanan maupun perangkat merupakan persyaratan penting untuk mencapai tujuan dari sistem ini.

Menurut penelitian yang dilakukan oleh Bhatti, dkk (2014) pada penelitiannya yakni "*Service Discovery Protocol in Pervasive Computing: A Review*" menyatakan ada beberapa komponen penting yang termasuk syarat dari sistem pervasif ini yakni pendaftaran suatu layanan, penemuan layanan dan interaksi layanan yang ditemukan. Sehingga setiap protokol maupun mekanisme pervasif yang dikembangkan harus mengandung 3 komponen tersebut untuk mewujudkan suatu sistem yang memiliki model pervasif.

2.2.3 Arsitektur *Publish-Subscribe*

Arsitektur *publish-subscribe* merupakan salah satu jenis arsitektur yang berada didalam komunikasi sistem terdistribusi (Dian, 2017). Mekanisme komunikasi yang terjadi didalam arsitektur ini merupakan komunikasi tidak langsung, sehingga perlu ada sesuatu yang menjembatani untuk melakukan suatu komunikasi antara pengirim dan penerima. Ada 3 entitas utama yang digunakan sebagai syarat komunikasinya yakni *broker* sebagai perantara, *publisher* sebagai pihak penyedia informasi serta *subscriber* sebagai pihak yang membutuhkan informasi.

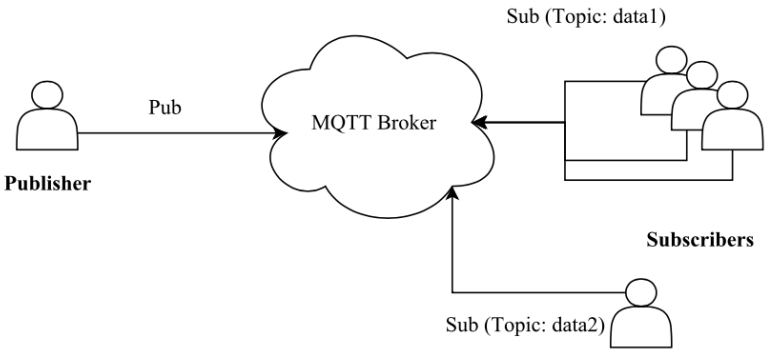
Subscriber dan *publisher* akan menjalin komunikasi berdasarkan topik. Topik merupakan identitas tertentu yang memiliki pendekatan yakni ketertarikan atas suatu informasi. *Subscriber* hanya menerima informasi dari broker sesuai dengan topik yang diminatinya, sehingga *publisher* tidak perlu mengetahui *subscriber* yang sedang aktif pada topik yang sama. Mekanisme yang ditawarkan pada arsitektur ini untuk sektor teknologi IoT memiliki banyak keuntungannya dibanding arsitektur sistem terdistribusi lainnya yakni informasi dapat dikirimkan secara efisien karena *subscriber* hanya menerima pesan sesuai dengan topik yang diminatinya (Kasyful, 2016).

Pengunaan arsitektur ini pada sistem yang berbasiskan sistem pervasif memungkinkan untuk melakukan komunikasi secara efisien, hal ini dibutuhkan dikarenakan mengingat jumlah entitas yang tidak terbatas dan sistem yang cendurung kompleks sehingga dapat menghemat daya pada setiap perangkat. Selain itu, arsitektur ini cocok untuk diimplementasikan pada perangkat IoT yang memungkin perangkat aktuator untuk dapat langsung menerima data tanpa melakukan *request* terlebih dahulu seperti pada arsitektur *client-server*.

2.2.4 MQTT

MQTT (*Message Queuing Telemetry Transport*) merupakan protokol komunikasi untuk model m2m (*Machine-to-Machine*) ataupun untuk perangkat IoT (*Internet of Things*). MQTT merupakan salah satu contoh protokol yang mengimplementasikan arsitektur *publish-subscribe* yang memungkinkan terjadinya komunikasi yang cepat dikarenakan arsitektur ini tidak memiliki mekanisme *request-response* yang umum digunakan pada protokol HTTP (*HyperText Transfer Protokol*) sehingga memungkinkan setiap pengguna akan mendapatkan informasi secara *realtime*.

Pada protokol MQTT ini terdapat suatu *middleware* yakni *broker* yang berfungsi sebagai pengendali pusat untuk mendistribusikan seluruh data yang terkoneksi ke dan dari *broker* tersebut. Pada Gambar 2.1 ditunjukan sistem sederhana yang biasanya digunakan untuk mengimplementasikan protokol MQTT, yakni *subscriber* hanya menerima informasi dengan berlangganan pada suatu topik, kemudian *publisher* mengirimkan informasi ke suatu topik, selama *subscriber* dan *publisher* berlangganan topik yang sama maka komunikasi akan selalu terjalin (Tantitharanukul, 2016).



Gambar 2.1 Sistem sederhana pada protokol MQTT

2.2.5 NodeMCU

NodeMCU sebuah papan mikrokontroler yang berbasiskan mikroprosesor ESP8266, dimana memiliki keunggulan yakni memiliki koneksitivitas *Wi-Fi* secara *on-board* dan memiliki kemampuan pengolahan dan penyimpanan yang lebih baik dibandingkan dengan Arduino UNO. Selain itu harga papan mikrokontroler ini cukup murah dan memiliki spesifikasi yang tinggi serta memiliki penggunaan daya yang cukup rendah sehingga dapat menghemat biaya dan penggunaan listrik.

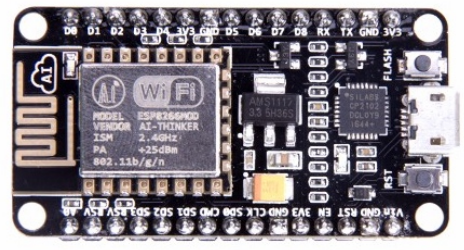
NodeMCU yang berbasiskan ESP8266 dirancang untuk perangkat yang bersifat *mobile*, perangkat *wearable* dan perangkat *Internet of Things* serta perangkat lainnya yang cenderung mengkonsumsi daya rendah. Papan mikrokontroler ini biasanya diaplikasikan untuk perangkat *smart home, home automation, industrial wireless control, IP camera, mesh network, wearable electronic, security id tags, Wifi position system beacon*, dll.

NodeMCU telah banyak didukung banyak bahasa pemrograman diantaranya adalah AT Command dan Lua sebagai bahasa resmi untuk *Node*MCU ini, lalu terdapat dukungan bahasa seperti C++ melalui *Library* pendukung yang perlu dipasang pada suatu kode editor yakni Platform.IO serta telah memiliki banyak dukungan *Library* yang dapat digunakan pada kode editor Arduino yakni, Arduino IDE sehingga untuk membuat program untuk papan mikrokontroler ini menjadi lebih mudah dan cepat. Tabel 2.1 adalah spesifikasi *Node*MCU dan Gambar 2.2 merupakan gambar *Node*MCU.

Tabel 2.1 Spesifikasi *Node*MCU

Microprocessor	ESP8266
Certificate	FCC/CE/TELEC/SRRC
Wi-Fi Protocol	802.11 b/g/n
Wi-Fi Security	WPA/WPA2
Wi-Fi Encryption	WEP/TKIP/AES
Frequency Range	2.4 – 2.5G
Peripheral	UART, SDIO, SPI, I2C, I2S, Remote Control, GPIO, PWM
Operating Voltage	3.0 – 3.6V
Operating Current	80mA
Operating Temperature	-40 – 125 ^{O}C
Network Protocol	IPv4, TCP/UDP, HTTP, FTP

Sumber: (Espressif, 2016)



Gambar 2.2 NodeMCU
Sumber: (Seedstudio, 2017)

2.2.6 Sensor dan Aktuator

Sensor merupakan perangkat yang melakukan konversi yang berasal dari kejadian di dunia fisik menjadi suatu sinyal yang dapat diukur dan juga dapat dianalisis atau yang bisa dikenal dengan tranducer yang mengubah energi dari suatu bentuk ke bentuk lainnya (Wiley, 2010). Aktuator merupakan suatu perangkat untuk menghasilkan masukan ke plant sesuai dengan sinyal kontrol sehingga sinyal umpan balik akan berkaitan dengan sinyal masukan acuan.

Penulis menggunakan sensor PIR (*Passive Infra Red*) serta sensor LDR (*Light Dependent Resistor*) serta aktuator berupa lampu pada penelitian ini. Sensor PIR digunakan untuk mendeteksi adanya suatu gerakan pada suatu lokasi uji. Fungsi ini cocok untuk diimplementasikan pada rumah cerdas untuk mengetahui ketersediaan seseorang didalam suatu rumah ataupun lokasi uji yang kemudian akan melakukan trigger tertentu terhadap aktuator.

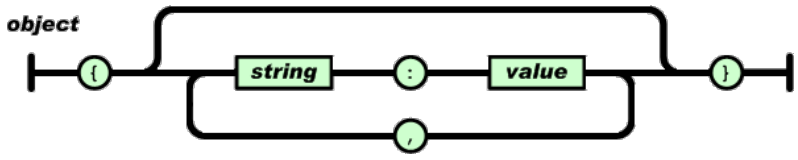
Sensor LDR pada penelitian ini digunakan untuk mendeteksi identitas cahaya pada suatu lokasi uji. Fungsi ini ssama halnya dengan sensor PIR yakni cocok untuk diimplementasikan pada konteks rumah cerdas untuk mengetahui kondisi diluar rumah apakah sedang gelap maupun terang sehingga kemudian akan menghasilkan suatu keputusan tertentu terhadap aktuator.

Selain objek lampu yang digunakan sebagai aktuator pada penelitian ini, terhadap relay. Relay merupakan komponen elektronik yang sama halnya seperti saklar yakni sebagai pusat kendali dari suatu aktuator. Relay ini akan memberikan keputusan berdasarkan data sensor yang diterima dari perangkat sensor sehingga menghasilkan keputusan tertentu baik menyalakan atau mematikan lampu.

2.2.7 JSON

JSON (*JavaScript Object Notation*) adalah format teks yang memfasilitasi pertukaran data terstruktur diantara semua bahasa pemorgraman (Ecma, 2013). JSON ini memiliki ukuran yang cenderung ringan, mudah dibaca dan ditulis oleh manusia serta mudah diterjemahkan dan dibuat oleh komputer. JSON juga merupakan format teks yang tidak bergantung dengan bahasa pemrograman apapun bahkan hampir seluruh bahasa pemrograman mendapat dukungan untuk format teks ini seperti C, C++, Java, JavaScript, Python, dll. Oleh karena itu, JSON sangat ideal sebagai format untuk pertukaran data.

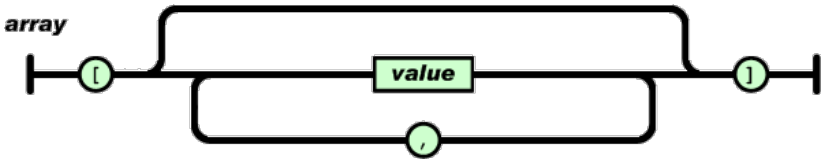
Salah satu bentuk pada format teks JSON adalah *Object*. Pada bentuk *JSON Object* ini tidak memiliki sepasang nama/nilai yang tidak berurutan serta memiliki karakteristik yang dimulai dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup), serta setiap nama diikuti dengan tanda : (titik dua) dan setiap pasangan nama dan nilai dipisahkan dengan tanda , (koma). Pada Gambar 2.3 menunjukan diagram sintaks untuk tipe *JSON Object*.



Gambar 2.3 Diagram sintaks JSON *Object*
Sumber: (ECMA, 2013)

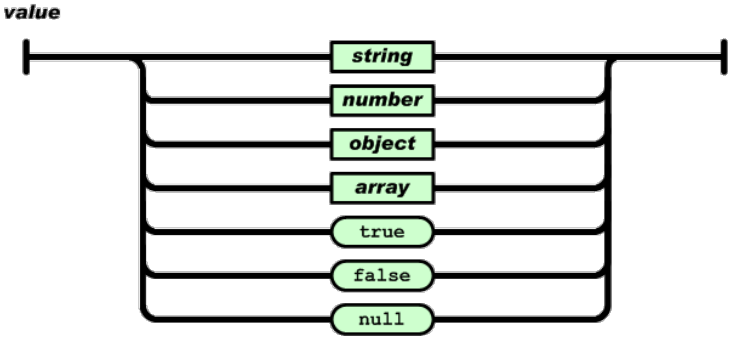
Pada bentuk lainnya yakni *JSON Array*, memiliki nilai yang berurutan yang diawali dengan index 0 (nol) dimana pada bentuk ini memiliki karakteristik yang dimulai dengan tanda [(kurung kotak terbuka) dan diakhiri dengan tanda]

(kurung kotak tertutup). Pada Gambar 2.4 menunjukkan diagram sintaks untuk tipe *JSON Array*.



Gambar 2.4 Diagram sintaks JSON *Array*
Sumber: (ECMA, 2013)

Kedua bentuk ini setiap pasangan nama/nilainya dipisahkan oleh tanda , (koma) dan memiliki nilai yang dapat berupa data *String*, Angka, *Boolean*, *Null* atau sebuah *Object* maupun *Array* yang dapat disusun secara bertingkat. Pada Gambar 2.5 menunjukkan diagram sintaks untuk nilai pada JSON.



Gambar 2.5 Diagram sintaks nilai pada JSON
Sumber: (ECMA, 2013)