



### **Hak cipta dan penggunaan kembali:**

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

### **Copyright and reuse:**

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

## BAB III

### METODOLOGI PENELITIAN

#### 3.1. Fitur dan Spesifikasi Penelitian

Fitur dan spesifikasi dari *Integrated Smart Lamps* dengan Jaringan Mesh ini adalah sebagai berikut:

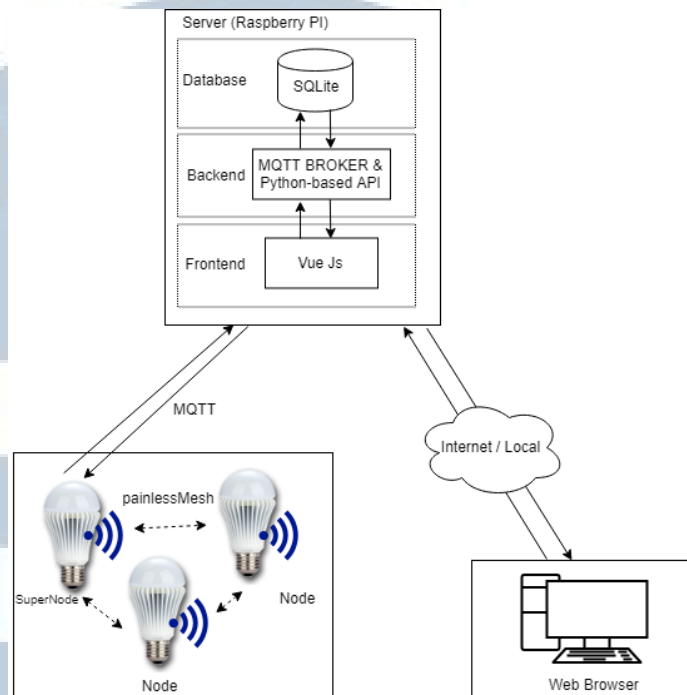
- 1) Pengguna dapat mengakses lampu pintar melalui website untuk mengubah pola atau melakukan *push* secara langsung.
- 2) Lampu pintar dapat menampilkan pola berupa tulisan pada waktu-waktu yang telah ditentukan oleh pengguna.
- 3) Pengguna dapat mengatur kecepatan perubahan pola pada lampu.
- 4) Antar lampu terintegrasi dan dapat saling berkomunikasi secara periodik agar waktu nya dapat berjalan secara bersama-sama.
- 5) Lampu yang baru terhubung akan secara otomatis mendapatkan perintah dari server.
- 6) Pola yang ada pada lampu berupa angka (0-9), huruf (A-Z) dan beberapa simbol lainnya pada matriks 5 x 5, pola ini didapatkan dari database yang diakses melalui API.

### 3.2. Studi Literatur

Pada tahap ini, penelitian dilakukan dengan mengumpulkan dan mempelajari referensi-referensi yang mendukung dan mempunyai hubungan dengan penelitian ini. Referensi berupa artikel, jurnal ilmiah, buku, dan *paper* yang berkaitan dengan penelitian ini dan, tulisan ilmiah yang menyerupai, serta website yang mendukung untuk dijadikan pedoman penelitian, seperti cara kerja *painlessMesh*, protokol MQTT, penggunaan ESP8266 beserta WeMos, *python-based API*, dan React JS.

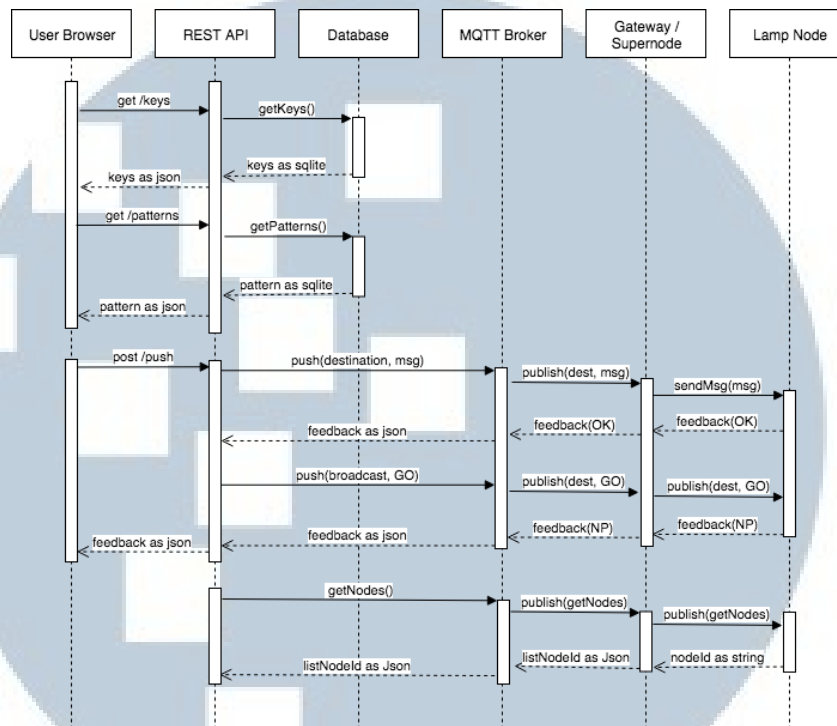
### 3.3. Perancangan Sistem Keseluruhan

Gambar 3.1 merupakan diagram rancangan sistem pada penelitian ini secara keseluruhan. Pada sistem ini, node pada lampu berkomunikasi satu dengan lain menggunakan mesh. Kemudian salah satu node ini bertugas sebagai *gateway* yang terhubung dengan server MQTT. Pengguna dapat mengakses website yang dijalankan oleh python server, kemudian website dapat mengakses server melalui REST API yang kemudian server menyimpan data pada *database*. Server kemudian mengirimkan perintah-perintah ke node-node lampu melalui *gateway* dengan protokol MQTT secara periodik dan sesuai jadwal yang dibuat pengguna melalui MQTT broker yang ada pada server menggunakan Paho, MQTT broker dari python.



Gambar 3.1 Block Diagram Sistem Keseluruhan

Alur sistem dapat dilihat pada Gambar 3.2, yaitu sequence diagram dari sistem yang dibuat. Pada gambar tersebut dapat terlihat bahwa server dapat memerintahkan broker untuk menjalankan beberapa fungsi melakukan publish fungsi ke node, maupun meminta daftar node yang terhubung dalam jaringan.



Gambar 3.2 Sequence Diagram Sistem Keseluruhan

Perangkat lunak pada server akan terbagi menjadi 3, yaitu *front-end* untuk menangani website, python yang berfungsi sebagai server sekaligus API sebagai kontroler yang menghubungkan dengan database, dan broker yang berfungsi untuk menangani pesan dari MQTT. Front-end dibuat dengan menggunakan JavaScript Framework VueJS. Sedangkan untuk API dan broker menggunakan python 3.7.0. Pengguna dapat mengakses server melalui website. Website akan mengakses database untuk mendapatkan *keys* pada setiap pola, setelah itu server akan mengambil data pada database dan mengembalikannya dalam bentuk JSON. Website kemudian mengelola data tersebut agar dan menampilkannya dalam bentuk pola matriks 5 x 5.

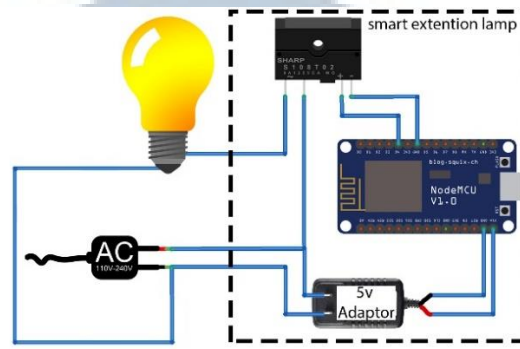
Setelah user memasukkan input, website akan melakukan *request* ke server dengan *method put* yang berisi input text dan juga *speed*. Setelah itu server akan melakukan save ke database dan melakukan *publish* ke setiap node sesuai polanya. Setelah node menerima perintah, node tidak langsung menjalankan pola yang diberikan, tetapi node akan menyimpan pola tersebut dan mengirimkan ACK ke server untuk memastikan bahwa setiap node sudah menerima perintah dari server. Setelah server menerima ACK dari seluruh node, server akan langsung memerintahkan node untuk menjalankan perintah yang sudah disimpan oleh node, dan node menjalankan pola tersebut.

### 3.4. Perancangan Perangkat Keras

Rancang bangun sistem ini membutuhkan 25 lampu pintar, dan setiap lampu pintar akan disusun seperti pada *block diagram* rancang bangun Gambar 3.3 berikut, yang terdiri atas:

- 1) Satu buah pengendali mikro WeMOS D1.
- 2) Satu buah Solid State Relay.
- 3) Satu buah adaptor 220 volt ke 5 volt.
- 4) Satu buah lampu LED 5 watt bohlam besar merk APA.

U N I V E R S I T A S  
M U L T I M E D I A  
N U S A N T A R A



Gambar 3.3 Block Diagram Rancang Bangun Lampu Pintar

NodeMCU atau WeMos secara *default* akan mengirimkan sinyal menyalakan ke Solid State Relay agar lampu menyala terus-menerus, sehingga apabila tidak ingin digunakan untuk menjalankan perintah lampu dapat digunakan seperti lampu biasa. Namun setiap kali pengendali mikro tersebut menerima perintah, sinyal akan dikirim sesuai perintah yang diberikan oleh sistem.

### 3.5. Perancangan Perangkat Lunak Pengendali Mikro

Pengendali mikro yang digunakan adalah NodeMCU dan WeMos. Setiap pengendali mikro akan dapat mengirimkan “hello” yang berfungsi sebagai *keep alive* agar koneksi tetap terjaga antar satu node dengan node lainnya. Antar pengendali mikro berkomunikasi berdasarkan jaringan mesh sederhana yang dibuat dengan menggunakan *library* *painlessMesh*.

Pengendali mikro pada node ini hanya berfungsi menerima perintah pesan dari server yang dikirim melalui *supernode* atau *gateway*. Sedangkan *supernode* berfungsi untuk mengirimkan semua pesan yang diterima dan diteruskan ke server, maupun perintah dari server ke setiap node. Setiap node diidentifikasi

menggunakan *chip\_id()* yang bersifat unik dan juga dari posisi node tersebut. Node – node pada lampu tersebut kemudian akan membaca pesan yang diterima dan mengidentifikasi pesan tersebut. Beberapa pesan perintah yang dapat dijalankan oleh setiap node dapat dilihat pada Tabel 3.1.

Tabel 3.1 Daftar Perintah yang Diterima oleh Node.

Nama	Format	Keterangan
PUSH	PUSH [PATTERN]	PUSH perintah ke setiap node untuk menjalankan pola. PATTERN berupa input 1/0 yang akan dijalankan secara berulang. Setelah menerima PUSH, node akan mengirimkan ACK ke server.
PLAY PATTERN	GO	Menjalankan perintah yang sudah diberikan setelah semua node mengirimkan ACK.
ON / STOP	ON  STOP	Memerintahkan node untuk menghentikan perintah yang sedang dijalankan dan menyalakan lampu secara terus menerus.
OFF	OFF	Memerintahkan node untuk menghentikan perintah yang sedang dijalankan dan mematikan lampu.
SET POSITION	SETP[POS]	Memerintahkan node untuk mengubah posisinya saat ini. POS berupa angka sebanyak dua <i>digit</i> , contoh 01, 05, 14, dsb.
SET SPEED TRANSITION	SETSP [SPEED]	Memerintahkan node untuk mengubah kecepatan transisinya. SPEED berupa angka antara 1 sampai dengan 9. Dengan 1 adalah 1 detik, dan berkurang 100 ms setiap kenaikan 1.



TEST	TEST	Memerintahkan node untuk mengubah kondisi lampu saat ini, apabila mati, maka akan menyala, apabila menyala, maka akan mati. Berfungsi untuk mencari posisi lampu.
------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------

Selanjutnya pada pengendali mikro yang berfungsi sebagai *supernode*, berfungsi untuk meneruskan semua perintah yang dikirimkan oleh server ke setiap node melalui MQTT dengan topik MQTT tertentu. Topik yang digunakan oleh server dapat dilihat pada Tabel 3.2.

Tabel 3.2 Daftar Topik MQTT yang Digunakan oleh Server.

Topik	Pengirim	Keterangan
to/broadcast	Server	Server mengirimkan perintah ke semua node.
to/[id]	Server	Server mengirimkan perintah ke spesifik node.
to/gateway	Server	Server mengirimkan perintah ke gateway. Contoh perintah adalah “getNodes”, yaitu meminta list semua nodes yang terdaftar.
from/gateway	gateway	Hasil balasan dari “getNodes” server dikirimkan melalui topik from/gateway dalam bentuk JSON.
from/[id]	gateway	Pesan yang dikirimkan oleh setiap nodes berdasarkan id, dikirimkan oleh gateway dengan topik berdasarkan <i>chip_id()</i> .

### 3.6. Perancangan Perangkat Lunak Server

Tahapan selanjutnya adalah perancangan perangkat lunak yang terdapat pada server. Ada tiga sistem besar yang dimiliki oleh server, yaitu VueJS sebagai *front-end*, Python-based API sebagai *back-end*, dan MQTT broker sebagai server MQTT.

VueJS berfungsi sebagai antarmuka untuk pengguna agar dapat berkomunikasi dengan sistem. VueJS menampilkan pola yang ada pada database ke user, dan user dapat mengubah beberapa pola tersebut. VueJS mengambil data pada server melalui REST API yang dibuat dengan menggunakan bahasa Python seperti pada Tabel 3.3

Tabel 3.3 Daftar Python-based API yang Digunakan.

Route	Method	Body	Keterangan
/patterns	get	-	Meminta seluruh pola yang ada pada database.
/pattern/<int>	get	-	Meminta pola pada waktu spesifik
/pattern/<int>	put	patterns	Mengubah pola yang ada pada database.
/pattern/<int>	delete	-	Menghapus pola yang ada pada database.
/keys	get	-	Meminta daftar pemetaan pola lampu 5 x 5.
/register	post	username, password	Melakukan registrasi pengguna
/auth	post	username, password	Melakukan login dengan menggunakan Json Web Token.

### 3.7. Dokumentasi

Dokumentasi dilakukan dengan memberi penjelasan pada program yang dibuat berupa penjelasan-penjelasan dan fungsi-fungsi dari variabel. Dokumentasi diperlukan agar aplikasi dapat dimahami dengan mudah bagi penulis maupun pembaca serta dengan tujuan agar peneliti lain dapat lebih mudah memahami apabila ingin mengembangkan penelitian ini.

