




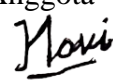



UNIVERSITAS DIPONEGORO – FAKULTAS TEKNIK  
DEPARTEMEN TEKNIK ELEKTRO

Jl. Prof. H. Soedarto, SH, Tembalang, Semarang 50275

Telp/Faks. (024)-7460057 e-mail: departemen@elektro.undip.ac.id

**Dokumen Pengembangan Produk**  
**Lembar Sampul Dokumen**

Judul Dokumen	<b>TUGAS AKHIR:</b> <b>Rancang Bangun Sistem Keamanan Kunci Pintu</b> <b>Gedung Berbasis <i>Internet of Things</i></b>
Jenis Dokumen	<b>DESAIN</b> <small>Catatan: Dokumen ini dikendalikan penyebarannya oleh Dept. Teknik Elektro Undip</small>
Nomor Dokumen	<b>B300-01-TA2223.2.19012</b>
Nomor Revisi	<b>01</b>
Nama File	<b>B300-2-TA2223</b>
Tanggal Penerbitan	<b>8 September 2023</b>
Unit Penerbit	<b>Departemen Teknik Elektro Undip</b>
Jumlah Halaman	<b>25</b> (termasuk lembar sampul ini)

Data Pengusul				
Pengusul	Nama	<b>Henric Dhiki Wicaksono</b>	Jabatan	Anggota
	NIM	21060119120011	Tanda Tangan	
	Nama	<b>Novi Dianasari</b>	Jabatan	Anggota
	NIM	21060119120039	Tanda Tangan	
	Nama	<b>Muhammad Khoiril Wafi</b>	Jabatan	Anggota
	NIM	21060119140133	Tanda Tangan	
Pembimbing Utama	Nama	<b>M. Arfan, S.Kom., M.Eng.</b>	Tanda Tangan	
Pendamping	Nama	<b>Imam Santoso, S.T., M.T.</b>	Tanda Tangan	
	NIP	197012031997021001		

## DAFTAR ISI

<b>1.</b>	<b>PENDAHULUAN .....</b>	<b>4</b>
1.1	Ringkasan Isi Dokumen .....	4
1.2	Aplikasi Dokumen.....	4
1.3	Referensi.....	4
1.4	Daftar Singkatan.....	5
<b>2.</b>	<b>SOLUSI DESAIN .....</b>	<b>6</b>
2.1	Solusi Desain 1 .....	6
2.2	Solusi Desain 2 .....	6
2.3	Solusi Desain 3 .....	7
<b>3.</b>	<b>PERANCANGAN.....</b>	<b>7</b>
3.1	Metode Verifikasi Akses .....	7
3.2	Perangkat Penguncian .....	9
3.3	Komunikasi Data Dua Arah .....	12
3.4	<i>Database</i> .....	14
3.5	<i>Backend API</i> .....	15
3.6	<i>Aplikasi Mobile</i> .....	17
3.7	Tampilan <i>Website</i> .....	20
<b>4.</b>	<b>VERIFIKASI .....</b>	<b>23</b>
4.1	Komunikasi WiFi dan <i>Bluetooth</i> .....	23
4.2	<i>Backend API Laravel</i> .....	24
4.3	Kode QR.....	25
<b>5.</b>	<b>PENUTUP .....</b>	<b>25</b>

## Catatan Sejarah Perbaikan Dokumen

VERSI, TGL, OLEH	PERBAIKAN
01, 8 September 2023, oleh Henric Dhiki Wicaksono, Novi Dianasari, dan Muhammad Khoiril Wafi.	<i>Draft</i> Dokumen B300

## **1. PENDAHULUAN**

### **1.1 Ringkasan Isi Dokumen**

Dokumen ini berisi tentang uraian desain untuk merealisasikan produk berupa “Sistem Keamanan Kunci Pintu Gedung Berbasis IoT”. Uraian desain terdiri dari pemilihan solusi yang digunakan dalam menyelesaikan permasalahan serta gambaran cara kerja dari semua fitur yang telah dijabarkan. Dokumen ini juga akan menjelaskan desain dari subsistem yang ada yaitu perangkat kunci pintu, sistem komunikasi data dua arah, *database*, *server*, *website*, dan aplikasi *mobile*. Desain subsistem meliputi pemilihan kerangka kerja serta algoritma yang akan digunakan dalam mengimplementasikan sistem tersebut.

Dokumen ini selanjutnya akan digunakan sebagai acuan dalam proses implementasi “Sistem Keamanan Kunci Pintu Gedung Berbasis IoT” serta sebagai bahan evaluasi pada proses pengembangan selanjutnya.

### **1.2 Aplikasi Dokumen**

Dokumen ini digunakan dalam proses pengembangan produk “Rancang Bangun Sistem Keamanan Kunci Pintu Gedung Berbasis IoT” untuk:

1. Menjelaskan pemilihan desain sesuai dengan subsistem yang ada.
2. Gambaran mengenai desain dan cara kerja sistem.
3. Menjelaskan standar-standar yang digunakan.
4. Referensi komponen atau *library* yang digunakan.
5. Menjadi catatan proses pengerjaan dan revisi.

Dokumen B300 ini diajukan kepada dosen pembimbing tugas akhir dan tim tugas akhir Program Studi Sarjana Teknik Elektro Undip sebagai bahan penilaian tugas akhir.

### **1.3 Referensi**

- [1] I. Hermawan, D. Arnaldy, P. Oktivasari, and D. A. Fachrudin, “Development of Intelligent Door Lock System for Room Management Using Multi Factor Authentication,” vol. 16, no. 1, pp. 1–14, 2023.
- [2] K. Y. Sun, Y. Pernando, and M. I. Safari, “Perancangan Sistem IoT pada Smart Door Lock Menggunakan Aplikasi BLYNK,” *JUTSI (Jurnal Teknol. dan Sist. Informasi)*, vol. 1, no. 3, pp. 289–296, 2021, doi: 10.33330/jutsi.v1i3.1360.
- [3] C. Asiminidis, G. Kokkonis, and S. Kontogiannis, “Database Systems

Performance Evaluation for IoT Applications,” *SSRN Electron. J.*, no. November 2019, 2019, doi: 10.2139/ssrn.3360886.

- [4] N. A. Amir Hamzah, M. R. Abu Saad, W. Z. Wan Ismai, T. Bhunaeswari, and N. Z. Abd Rahman, “Development of A Prototype of An IoT Based Smart Home with Security System Flutter Mobile,” *J. Eng. Technol. Appl. Phys.*, vol. 1, no. 2, pp. 34–41, 2019, doi: 10.33093/jetap.2019.1.2.70.

## 1.4 Daftar Singkatan

**Tabel 1.1** Daftar singkatan.

SINGKATAN	ARTI
IoT	<i>Internet of Things</i>
WiFi	<i>Wireless Fidelity</i>
JSON	<i>Javascript Object Notation</i>
IOS	<i>iPhone Operating System</i>
QR-Code	<i>Quick Response Code</i>
HTTP	<i>Hypertext Transfer Protocol</i>
API	<i>Application Programming Interface</i>
ERD	<i>Entity Relationship Diagram</i>
MVC	<i>Model-View Controller</i>
PHP	<i>Hypertext Preprocessor</i>
SPA	<i>Single Page Application</i>
AJAX	<i>Asynchronous Javascript and XML</i>
DOM	<i>Document Object Model</i>
OTP	<i>One-Time Password</i>
RFID	<i>Radio Frequency Identification</i>
ESP	<i>Espressif</i>
MySQL	<i>My Structured Structured Query Language</i>

**Tabel 1.1** (lanjutan)

SINGKATAN	ARTI
HP	<i>Handphone</i>
IDE	<i>Integrated Development Environment</i>
URL	<i>Uniform Resource Locator</i>

## 2. SOLUSI DESAIN

Berdasarkan uraian dokumen B200 sebelumnya, dibutuhkan sebuah sistem yang dapat digunakan untuk mengelola kunci pintu gedung secara aman dan efisien dengan berbagai fitur yang dimiliki seperti pemantauan, pengaturan hak akses, kendali jarak jauh, dan lain sebagainya. Oleh sebab itu, diperlukan desain sistem yang dapat menjalankan semua fitur tersebut. Pemilihan desain didasarkan pada kebutuhan serta keamanan dari sistem untuk menjalankan semua fitur yang ada dengan menggunakan ilmu rekayasa yang terkait sehingga menghasilkan desain sistem yang aman dan efisien.

### 2.1 Solusi Desain 1

Solusi desain yang pertama yaitu menggunakan konsep IoT untuk mengelola semua kunci pintu yang sebelumnya berupa kunci fisik menjadi kunci digital. Pada setiap pintu dalam satu gedung akan terpasang sebuah perangkat kunci digital yang dapat dipantau serta dapat dikendalikan oleh semua pengguna gedung tersebut. Dengan menggunakan perangkat penguncian secara digital memungkinkan untuk melakukan pengelolaan kunci dengan lebih aman dan efektif.

### 2.2 Solusi Desain 2

Solusi desain yang kedua yaitu membangun sebuah layanan untuk mengendalikan sistem yang sudah dibangun oleh perangkat kunci digital. Sebuah *server* dapat menyediakan layanan untuk melakukan pemantauan serta memberikan perintah ke perangkat kunci pintu secara otomatis. Setiap perangkat kunci pintu akan terhubung ke *server* dengan menggunakan koneksi internet. *Server* juga dapat menentukan siapa saja yang diizinkan untuk mengakses pintu berdasarkan data yang tersimpan di dalamnya.

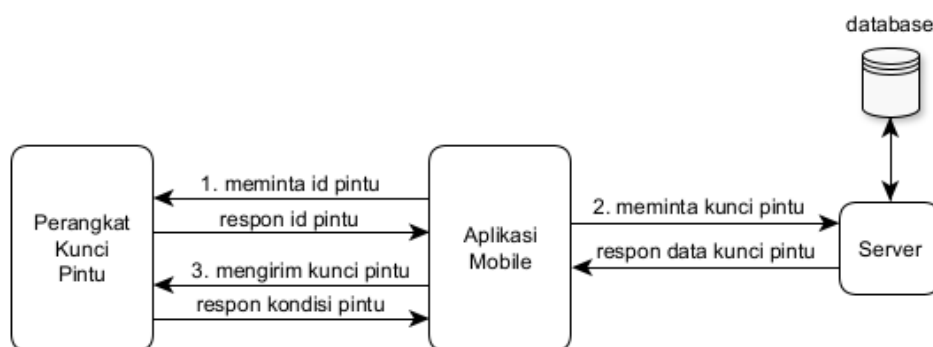
## 2.3 Solusi Desain 3

Solusi desain yang ketiga yaitu membuat sebuah tampilan sebagai media yang digunakan oleh pengguna gedung untuk berinteraksi dengan perangkat kunci pintu seperti membuka kunci atau melihat kondisi pintu secara jarak jauh. Sebuah aplikasi *mobile* dan *website* dapat digunakan sebagai antarmuka pengguna untuk berinteraksi dengan sistem. Adapun aplikasi *mobile* dapat digunakan untuk membuka atau memantau kondisi pintu dengan karakteristiknya yang ringkas dan mudah, sedangkan *website* digunakan untuk melakukan pengaturan sistem.

## 3. PERANCANGAN

### 3.1 Metode Verifikasi Akses

Setiap pintu pada gedung akan dikunci menggunakan perangkat kunci pintu digital sehingga dapat digunakan untuk melakukan verifikasi akses kepada semua pengguna yang ingin mengakses ruangan tersebut. Proses verifikasi akses dapat dilaksanakan dengan menggunakan berbagai macam metode. Penelitian yang dilakukan oleh Indra Hermawan dkk [1] tentang autentikasi pada kunci pintu pintar, RFID dapat digunakan untuk melakukan autentikasi kunci pintu. Namun, metode tersebut tidak menyediakan solusi yang tepat untuk menyelesaikan permasalahan, dikarenakan RFID hanya akan menggantikan tempat dari kunci fisik. Oleh karena itu, pada penelitian ini dikembangkan metode lain untuk melakukan autentikasi akses yaitu dengan metode *Access Control List* (ACL). Metode tersebut digunakan untuk menentukan apakah pengguna diizinkan untuk masuk ke ruangan atau tidak. Diagram proses verifikasi akses yang akan digunakan pada sistem ini dapat dilihat pada Gambar 3.1 di bawah ini.



**Gambar 3.1** Diagram proses verifikasi akses.

Pada Gambar 3.1 di atas terlihat bahwa proses verifikasi akses melibatkan tiga bagian yaitu perangkat kunci pintu, aplikasi *mobile*, dan *server* menggunakan verifikasi tiga langkah dengan penjelasan sebagai berikut:

1. Meminta identitas pintu

Untuk membuka kunci pintu tentunya perlu mengetahui pintu mana yang akan dibuka. Identitas pintu akan membedakan pintu satu dengan lainnya sehingga setiap pintu akan memiliki identitas yang unik. Identitas pintu berupa kode QR sehingga memudahkan aplikasi *mobile* untuk mengenali pintu dengan memindai kode QR tersebut.

2. Meminta kunci pintu

Setelah mendapatkan identitas pintu, aplikasi *mobile* akan meminta kode kunci pintu dengan mengirimkan permintaan ke *server* disertai dengan identitas pengguna. Kemudian, *server* akan memeriksa data akses pengguna dengan menggunakan data identitas pintu dan pengguna. Jika aksesnya cocok, maka *server* akan mengembalikan kode kunci pintu yang digunakan untuk membuka pintu.

3. Mengirim kunci pintu

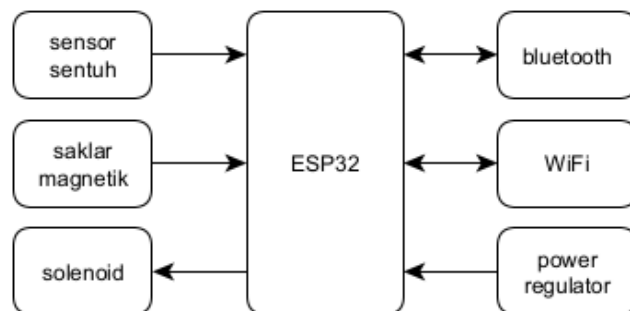
Setelah mendapatkan kode kunci pintu, aplikasi *mobile* akan mengirimkan kode kunci tersebut melalui koneksi *bluetooth* untuk membuka kunci pintu.

Berdasarkan penjelasan metode autentikasi di atas, proses untuk membuka kunci pintu yaitu menggunakan koneksi *bluetooth* dengan cara memindai kode QR statis via aplikasi *mobile* yang telah disediakan. Hal tersebut dimungkinkan adanya penduplikasian kode QR. Dengan adanya kekurangan tersebut, penggunaan *bluetooth* dapat dipastikan bahwa pengguna benar-benar memindai kode QR di area atau tepat di depan pintu karena karakteristik *bluetooth* yang memiliki batas jarak koneksi yang dekat. Adapun untuk membuka kunci pintu maka pengguna menggunakan kode kunci pintu dinamis. Kode tersebut akan berubah seiring dengan perubahan kondisi pintu sehingga proses autentikasi benar-benar aman dengan menggunakan kode kunci pintu yang dinamis.



### 3.2 Perangkat Penguncian

Untuk menjalankan proses penguncian secara digital, pada setiap pintu tentunya membutuhkan sebuah alat yang dapat digunakan untuk mengunci dan memantau kondisi pintu. Pada penelitian yang dilakukan oleh Kaleb Yefune Sun dkk [2] tentang sistem kunci pintu menggunakan BLYNK, ESP32 dapat digunakan untuk mengendalikan kunci pintu serta melakukan pemantauan kondisi pintu. Diagram perangkat penguncian dapat dilihat pada Gambar 3.2 di bawah ini.

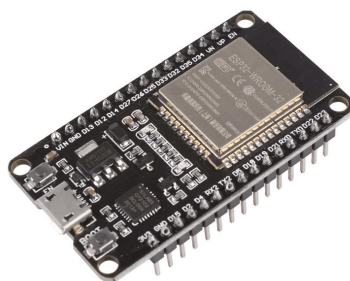


**Gambar 3.2** Diagram perangkat penguncian.

Pada Gambar 3.2 di atas terlihat bahwa perangkat penguncian menggunakan ESP32 sebagai komponen utama dan beberapa komponen tambahan sebagai berikut:

1. ESP32

ESP32 menjadi komponen utama di dalam perangkat penguncian. ESP32 merupakan sebuah mikrokontroler sehingga dapat digunakan untuk mengatur dan memantau kondisi pintu dengan mengolah data yang diterima dari sensor dan aktuator.

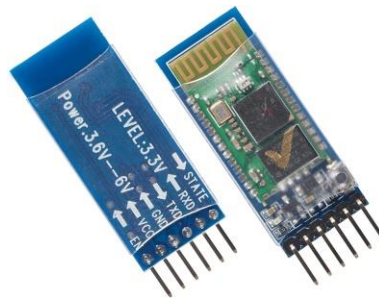


**Gambar 3.3** ESP32 development board.

Gambar 3.3 merupakan sebuah papan pengembangan ESP32. Pada papan pengembangan tersebut terdapat pin-pin digital yang dapat digunakan untuk membaca sensor atau menggerakkan *actuator*. Pada papan pengembangan tersebut juga terdapat modul *programmer* yang digunakan untuk menjalankan program pada proses pembuatan kode program.

## 2. *Bluetooth*

Proses autentikasi akses pengguna akan menggunakan koneksi *bluetooth* untuk mengirim kode kunci. Oleh karena itu, perangkat kunci pintu harus mempunyai modul komunikasi *bluetooth* sebagai jalur komunikasi untuk berkomunikasi dengan aplikasi *mobile*. Modul *bluetooth* yang digunakan harus dapat berkomunikasi dengan ESP32 sebagai kontroler utama. Adapun perangkat penguncian ini menggunakan modul HC-05 sebagai modul *bluetooth* dengan komunikasi serial yang didukung oleh ESP32.



**Gambar 3.4** Modul *bluetooth* HC-05.

Gambar 3.4 di atas merupakan modul *bluetooth* HC-05. Modul tersebut dapat berkomunikasi dengan perangkat *smartphone* melalui koneksi tanpa kabel (nirkabel). Pada modul *bluetooth* tersebut juga terdapat pin-pin yang digunakan untuk berkomunikasi dengan mikrokontroler melalui koneksi serial.

## 3. *WiFi*

Perangkat penguncian perlu terhubung dengan *server* untuk mengirimkan kondisi pintu atau menerima perintah dari jarak jauh. Oleh karena itu, ESP32 membutuhkan modul komunikasi yang dapat terhubung dengan internet. Pada ESP32 sudah tersedia modul WiFi yang dapat digunakan untuk terhubung ke internet sehingga tidak perlu menambah komponen WiFi eksternal.

#### 4. Saklar magnetik

Perangkat penguncian harus dapat memantau kondisi pintu untuk selanjutnya dilaporkan ke *server*. Sebuah sensor diperlukan untuk melakukan tugas pemantauan. Sebuah sensor magnetik dapat digunakan untuk mendeteksi kondisi pintu terbuka atau tidak dengan memanfaatkan medan magnet.



**Gambar 3.5** Saklar magnetik.

Gambar 3.5 di atas merupakan sebuah saklar magnetik. Sebuah saklar magnetik terdiri dari dua bagian yaitu bagian magnet dan saklar. Pada saat keduanya berdekatan maka saklar akan tertutup dan saat keduanya terpisah maka saklar akan terbuka. Oleh karena itu, dapat digunakan untuk mendeteksi kondisi pintu sedang terbuka atau tertutup.

#### 5. Solenoid

Sebuah aktuator diperlukan untuk mengunci pintu secara fisik. Solenoid dapat digunakan untuk mengunci pintu fisik dengan kendali berupa arus listrik. Dengan menggunakan solenoid, kendali penguncian dapat dilakukan secara digital dengan mengirimkan arus listrik.



**Gambar 3.6** Solenoid.

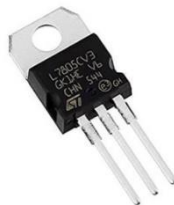
Gambar 3.6 di atas merupakan sebuah solenoid. Solenoid menggunakan prinsip elektromagnetik untuk menarik pin kunci sehingga dapat digunakan untuk mengunci pintu dengan memberikan arus listrik yang sesuai.

#### 6. Sensor sentuh

Dalam kondisi normal, sebuah solenoid akan selalu mengunci pintu dan solenoid hanya dapat membuka kunci pintu dalam waktu yang relatif singkat atau tidak dapat membuka kunci pintu terus-menerus. Oleh karena itu, diperlukan mekanisme untuk mendeteksi pengguna yang membuka pintu untuk mengaktifkan solenoid. ESP32 menyediakan sebuah sensor sentuh yang dapat digunakan untuk mendeteksi sentuhan pada gagang pintu sehingga informasi tersebut dapat digunakan untuk mengaktifkan solenoid.

#### 7. *Power regulator*

Untuk dapat beroperasi tentunya perangkat penguncian memerlukan arus listrik. Setiap komponen yang digunakan pada perangkat penguncian mungkin memerlukan voltase yang berbeda sehingga diperlukan untuk menyediakan voltase yang sesuai dengan kebutuhan setiap komponen.



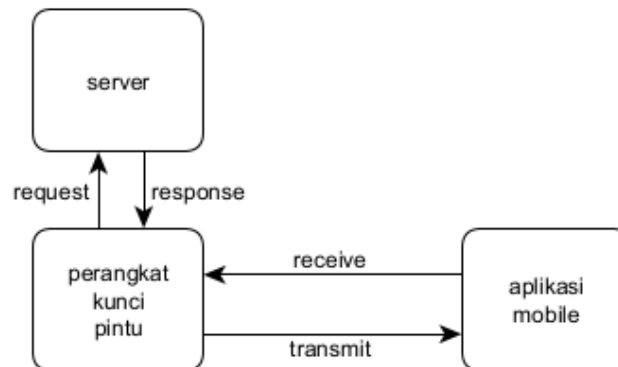
**Gambar 3.7** Regulator 5 volt.

Gambar 3.7 di atas merupakan sebuah regulator untuk menurunkan tegangan masukan menjadi 5 volt. Perangkat penguncian menggunakan tegangan masukan sebesar 12 volt. Hal tersebut sesuai dengan voltase kerja dari solenoid, sedangkan ESP32 bekerja pada voltase 5 volt sehingga diperlukan sebuah regulator untuk menurunkan tegangan 12 volt menjadi 5 volt.

### 3.3 Komunikasi Data Dua Arah

Perangkat penguncian perlu berkomunikasi dengan perangkat lain seperti *smartphone* dan *server*, baik itu untuk menerima perintah atau mengirimkan respon. Oleh

karena itu, perangkat penguncian membutuhkan komunikasi data dua arah untuk mengirim dan menerima data yang akan digunakan pada proses selanjutnya.



**Gambar 3.8** Komunikasi data dua arah.

Gambar 3.8 di atas menampilkan diagram komunikasi yang dilakukan oleh perangkat kunci pintu dengan *server* dan aplikasi *mobile*. Komunikasi yang dilakukan oleh perangkat kunci pintu terbagi menjadi dua yaitu komunikasi dengan *server* dan komunikasi dengan aplikasi *mobile* dengan penjelasan sebagai berikut:

1. Komunikasi dengan *server*

Perangkat kunci pintu berkomunikasi dengan *server* untuk menerima perintah secara jarak jauh dan untuk melaporkan kondisi pintu secara *realtime*. Untuk menjalin komunikasi tersebut, perangkat kunci pintu menggunakan protokol HTTP dengan metode API. Perangkat kunci pintu menggunakan API yang telah disediakan oleh *server* untuk melaporkan kondisi pintu.

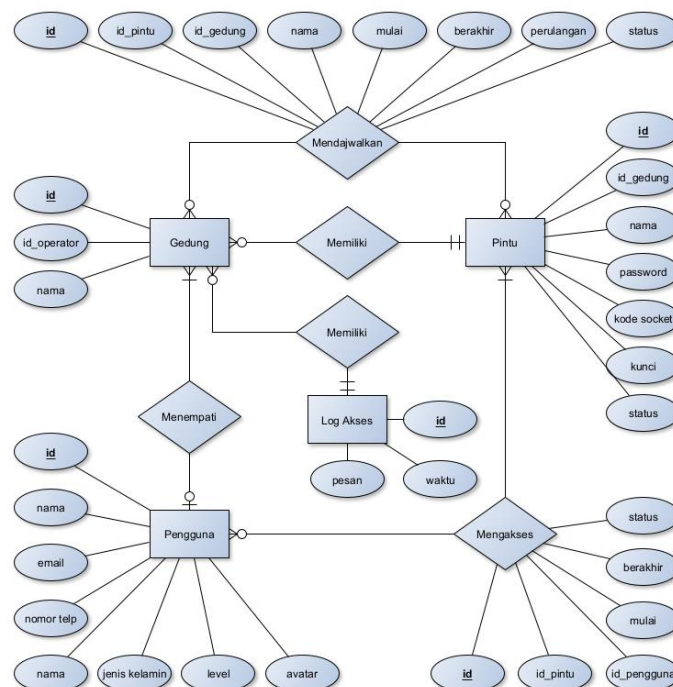
Untuk menerima perintah dari *server*, metode yang digunakan sedikit berbeda dikarenakan perangkat kunci pintu bertindak sebagai *client*. Untuk menerima perintah dari *server*, perangkat kunci pintu menggunakan protokol *websocket*. *Websocket* digunakan untuk menjalin koneksi dengan *server* dan koneksi tersebut akan terus terjalin. Dengan menggunakan koneksi tersebut, perangkat kunci pintu dapat menerima *event* dari *server* meskipun perangkat kunci pintu tidak melakukan permintaan data. Dengan menggunakan metode tersebut, perintah dapat dikirimkan secara langsung tanpa menunggu *polling* yang dilakukan oleh perangkat kunci pintu.

## 2. Komunikasi dengan aplikasi *mobile*

Untuk berkomunikasi dengan aplikasi *mobile*, perangkat kunci pintu menggunakan koneksi *bluetooth* yang umumnya disediakan oleh semua perangkat *smartphone*. Dengan menggunakan *bluetooth* maka proses autentikasi hanya bisa dilakukan di dekat perangkat kunci pintu dan juga memberikan keuntungan bahwa perangkat kunci pintu tetap dapat bekerja meskipun tanpa koneksi internet.

### 3.4 Database

*Database* digunakan untuk menyimpan data yang digunakan dalam sistem penguncian pintu gedung seperti data pengguna, data pintu, data akses, dan data lainnya. *Database* yang digunakan harus dapat menunjang kinerja sistem dengan karakteristik respon yang cepat dan pengelolaan data terstruktur. Berdasarkan penelitian [3] yang membandingkan kinerja dari berbagai tipe dan jenis *database*, didapatkan bahwa penggunaan MySQL menunjukkan hasil kinerja yang bagus dalam hal waktu eksekusi permintaan. Dengan sistem penyimpanan data bersifat relasional dan terstruktur maka MySQL dapat diterapkan pada sistem penguncian pintu gedung ini.

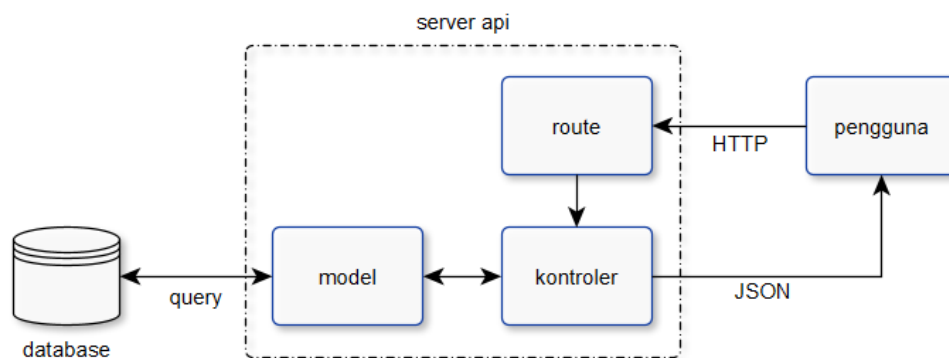


Gambar 3.9 ERD database.

Pada Gambar 3.9 di atas terlihat relasi antar entitas di dalam *database* sistem penguncian pintu gedung. Pada *database* sistem penguncian pintu gedung terdapat beberapa entitas seperti pengguna, pintu, gedung dan *log* akses. Entitas tersebut digunakan untuk menyimpan data yang akan digunakan dalam pengelolaan sistem dengan isi data sesuai dengan atribut dari masing-masing entitas. Di dalam sistem *database* juga terdapat relasi. Relasi menunjukkan hubungan antar entitas seperti menempati, memiliki, menjadwalkan, dan mengakses. Khusus untuk relasi menjadwalkan dan mengakses pada Gambar 3.9 terlihat bahwa relasi tersebut memiliki atribut yang menjadi penghubung antara dua entitas yang berbeda. Pada umumnya, kondisi tersebut menunjukkan kardinalitas *many-to-many* yang bertindak sebagai entitas pivot untuk menghubungkan relasi *many-to-many* dari dua entitas lainnya.

### 3.5 Backend API

Sebuah API dibutuhkan sebagai layanan komunikasi yang digunakan oleh perangkat kunci pintu dan aplikasi *mobile* untuk berkomunikasi dengan *server*. Selain terdapat *website* yang digunakan sebagai antarmuka utama dalam mengelola sistem ini, juga terdapat aplikasi berbasis *mobile* yang digunakan untuk menunjang kinerja dari sistem terutama pada bagian yang membutuhkan teknologi yang belum disediakan oleh *browser* seperti koneksi *bluetooth* dan pindai kode QR menggunakan kamera.



**Gambar 3.10** Diagram *backend* API.

Pada Gambar 3.10 di atas terlihat diagram *backend* API. *Backend* API dibangun menggunakan konsep MVC yang disediakan oleh laravel. Laravel merupakan sebuah kerangka kerja pengembangan *website* dan API dengan bahasa pemrograman PHP yang

menggunakan konsep MVC. Dengan menggunakan Laravel, proses pengembangan API dapat dilakukan dengan cepat serta menggunakan berbagai modul-modul yang telah tersedia seperti autentikasi, koneksi *database*, *query*, token, dan lain sebagainya. API akan menyediakan layanan berupa *endpoint* yang dapat diakses oleh perangkat kunci pintu maupun aplikasi *mobile*. *Endpoint* yang tersedia dapat dilihat pada Tabel 3.1 di bawah.

**Tabel 3.1** *Endpoint* API.

<b>Type</b>	<b>Endpoint</b>	<b>Auth</b>	<b>Keterangan</b>
POST	<i>/api/login</i>	-	<i>login</i> pengguna dan operator
POST	<i>/api/reset-password</i>	-	<i>reset password</i> menggunakan email
POST	<i>/api/verify-email</i>	<i>Sanctum</i>	verifikasi email
GET	<i>/api/logout</i>	<i>Sanctum</i>	<i>logout</i> pengguna dan operator
POST	<i>/api/update-profile</i>	<i>sanctum, verified</i>	<i>update profile</i> nama, email, dan lainnya
GET	<i>/api/avatar</i>	<i>sanctum, verified</i>	mengambil gambar avatar
POST	<i>update-avatar</i>	<i>sanctum, verified</i>	mengubah gambar avatar
POST	<i>/api/change-password</i>	<i>sanctum, verified</i>	mengubah <i>password</i> pengguna atau operator
GET	<i>/api/my-access</i>	<i>sanctum, verified</i>	mengambil daftar akses
GET	<i>/api/get-doors</i>	<i>sanctum, verified</i>	mengambil daftar pintu
GET	<i>/api/my-history</i>	<i>Sanctum, verified</i>	mengambil daftar riwayat akses
GET	<i>/api/verify-access/{door_id}</i>	<i>sanctum, verified</i>	verifikasi akses dari kode QR pintu
POST	<i>/api/remote-access</i>	<i>sanctum, verified</i>	membuka atau mengunci pintu jarak jauh
POST	<i>/door/login</i>	-	<i>login</i> perangkat kunci pintu
POST	<i>/door/register</i>	-	menambahkan perangkat kunci pintu baru



Tabel 3.1 (lanjutan)

Type	Endpoint	Auth	Keterangan
GET	/door/logout	Sanctum	logout perangkat kunci pintu
POST	/door/get-signature	Sanctum	mengambil kode <i>signature</i> <i>channel pusher</i>
POST	/door/update-status	Sanctum	update status pintu
POST	/door/alert	Sanctum	peringatan pada pintu

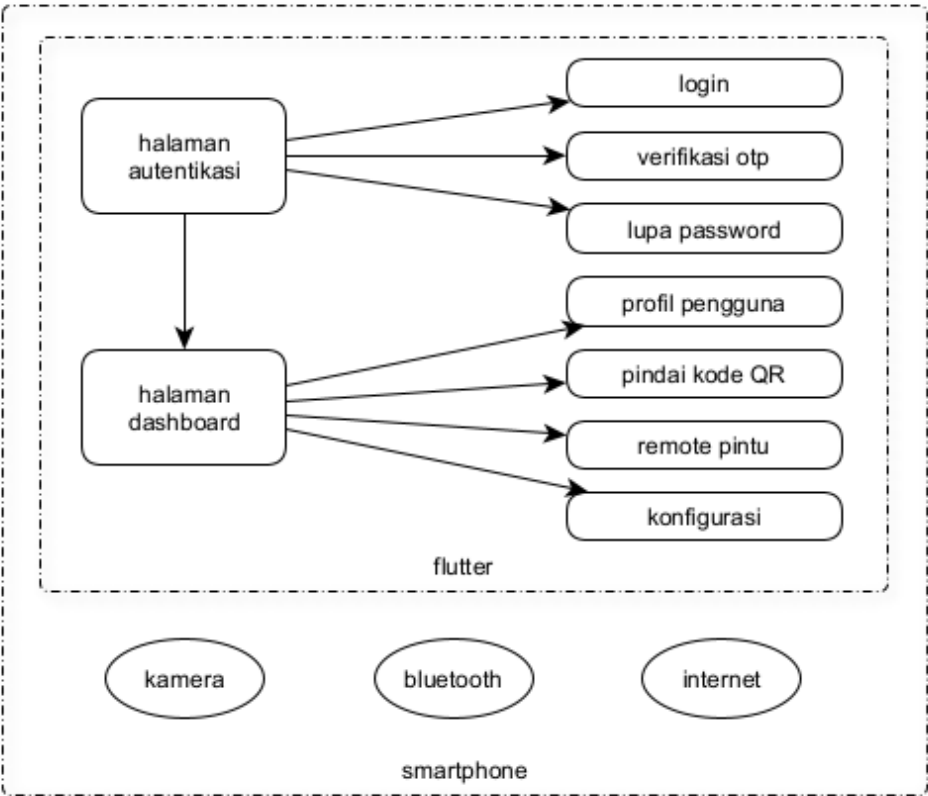
Dapat dilihat pada Tabel 3.1 di atas, API terbagi menjadi 2 bagian yang ditandai dengan awalan (*prefix*) “/api” dan “/door”. Awalan “/api” merupakan *endpoint* API yang khusus ditujukan untuk penggunaan aplikasi *mobile* pengguna dan operator, sedangkan awalan “/door” merupakan API yang ditujukan untuk perangkat kunci IoT. Dengan menggunakan API yang terpisah maka API akan semakin terorganisasi dengan baik.

Pada Tabel 3.1 juga terlihat bahwa beberapa *endpoint* memerlukan autentikasi *sanctum* dan *verified*. Autentikasi *sanctum* merupakan sebuah metode autentikasi berbasis token yang digunakan untuk mengamankan sumber daya API dari *client* dengan hanya mengizinkan pengguna yang sudah terautentikasi yang dapat mengakses API tersebut. Sedangkan, *verified* merupakan autentikasi tambahan yang digunakan untuk memastikan bahwa *client* (pengguna dan operator) sudah melakukan verifikasi email sehingga dapat meningkatkan keamanan API.

### 3.6 Aplikasi Mobile

Sebuah aplikasi *mobile* diperlukan untuk membuka kunci pintu dengan menggunakan kode QR. Aplikasi *mobile* akan menyediakan fitur koneksi *bluetooth* yang belum tersedia pada sebuah *website* sehingga proses verifikasi akses dapat berjalan dengan baik. Untuk menjalankan proses verifikasi akses dan fitur lainnya, maka aplikasi *mobile* yang dikembangkan harus memiliki beberapa kemampuan yang mendukung kinerja dari sistem penguncian seperti koneksi internet, koneksi *bluetooth*, serta pemindai kode QR. Pada penelitian yang dilakukan oleh Nur Asyiqin Bt. Amir Hamzah dkk [4] tentang perancangan rumah pintar berbasis IoT, kerangka kerja *Flutter* dapat digunakan dalam pengembangan aplikasi *mobile*. *Flutter* merupakan sebuah kerangka kerja untuk membuat aplikasi *mobile*. *Flutter* merupakan kerangka kerja *cross-platform* yang artinya aplikasi yang dihasilkan dapat dijalankan pada sistem operasi yang berbeda misalnya

Android dan IOS dengan menggunakan satu struktur program yang sama. Secara umum, aplikasi yang akan dikembangkan digunakan untuk berinteraksi dengan perangkat IoT misalnya untuk membuka kunci pintu, melihat kondisi pintu, dan melakukan pengaturan.



Gambar 3.11 Struktur aplikasi *mobile*.

Gambar 3.11 memperlihatkan struktur dari aplikasi *mobile* yang akan dikembangkan. Pada aplikasi *mobile* akan terdapat beberapa layar yang dapat digunakan untuk berinteraksi dengan sistem kunci pintu. Penjelasan mengenai setiap layar dapat dilihat pada Tabel 3.2 di bawah ini.

Tabel 3.2 Pembagian layar aplikasi *mobile*.

Nama	Autentikasi	Penjelasan
Login	-	Halaman autentikasi untuk <i>login</i> pengguna dan operator. Proses <i>login</i> menggunakan alamat email dan <i>password</i> yang sudah terdaftar.

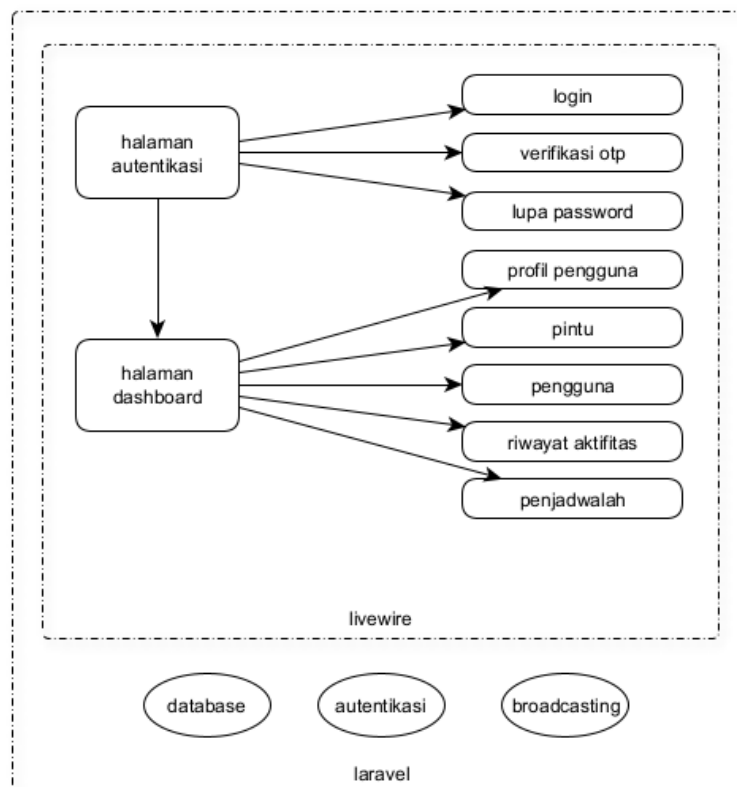
**Tabel 3.2** (lanjutan)

<b>Nama</b>	<b>Autentikasi</b>	<b>Penjelasan</b>
Verifikasi OTP	-	Jika email belum terverifikasi maka akan diarahkan ke halaman verifikasi. Verifikasi dengan menggunakan kode OTP yang dikirimkan ke email yang dimasukkan pada saat <i>login</i> .
Lupa <i>password</i>	<i>Login</i>	Pengguna aplikasi dapat memanfaatkan fitur ini jika pengguna kehilangan <i>password</i> , dengan cara memasukkan alamat email yang sesuai untuk menerima petunjuk lebih lanjut.
Profil pengguna	<i>Login</i>	Proses pengaturan pengguna dilakukan pada halaman profil pengguna. Pengguna aplikasi dapat menyesuaikan informasi diri seperti nama, nomor hp, dan foto profil.
Pindai kode QR	<i>Login</i>	Pengguna dapat menggunakan kode QR pada pintu untuk membuka kunci pintu. Pada halaman ini, pengguna memindai kode tersebut dan aplikasi akan meminta akses ke <i>server</i> secara otomatis.
<i>Remote</i> pintu	<i>Login, Operator</i>	Operator dapat membuka atau mengunci pintu secara jarak jauh melalui koneksi internet. Pada halaman ini juga menampilkan semua pintu dan kondisinya.
Konfigurasi	<i>Login, Operator</i>	Halaman ini digunakan untuk melakukan konfigurasi pada perangkat kunci pintu seperti mengatur kredensial WiFi atau mendaftarkan perangkat baru ke <i>server</i> .

Pada Gambar 3.11 juga terlihat bahwa aplikasi membutuhkan dukungan dari kamera, internet, dan *bluetooth*. Pada kerangka kerja *flutter*, penggunaan *peripheral* pada *smartphone* dapat dilakukan dengan menggunakan beberapa modul yang telah disediakan seperti modul *http* untuk menjalankan fungsi yang berkaitan dengan internet, modul *bluetooth\_serial* untuk menjalankan komunikasi *bluetooth* sederhana, dan *qr\_scanner* untuk menjalankan fungsi pemindaian menggunakan kamera.

### 3.7 Tampilan Website

Sistem keamanan kunci pintu gedung ini juga memerlukan sebuah *website* yang digunakan untuk melakukan pengaturan pada sistem tersebut. Dengan menggunakan *website* maka proses pengaturan menjadi lebih fleksibel. Proses pengaturan yang dilakukan meliputi pengelolaan pintu, pengguna, akses, penjadwalan, dan riwayat aktivitas. *Website* tersebut hanya bisa diakses oleh pengelola gedung sebagai operator yang menjadi penanggung jawab terhadap sistem keamanan yang dijalankan.



**Gambar 3.12** Struktur *website*.

Pada Gambar 3.12 memperlihatkan struktur dari *website* yang akan dikembangkan. Seperti halnya *backend API*, *website* dikembangkan dengan menggunakan kerangka kerja Laravel dengan memanfaatkan modul-modul yang telah disediakan seperti koneksi *database*, autentikasi, dan *broadcasting*. Seperti pada aplikasi *mobile*, *website* memiliki beberapa layar yang digunakan oleh operator untuk mengatur sistem seperti menambahkan pintu baru, membuat jadwal, menambahkan akses, menambahkan pengguna, dan lain sebagainya. Penjelasan fungsi setiap layar pada *website* dapat dilihat pada Tabel 3.3 di bawah ini.

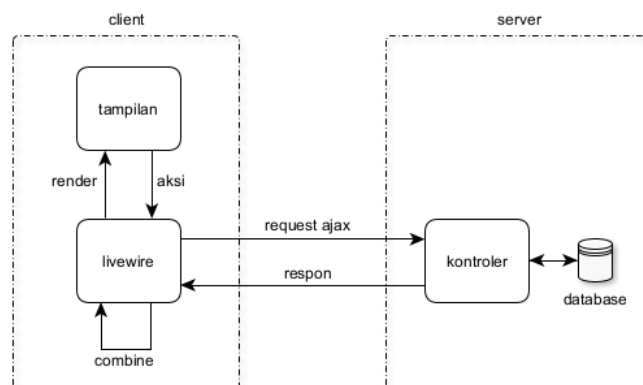
**Tabel 3.3** Pembagian layar *website*.

<b>Nama</b>	<b>Autentikasi</b>	<b>Penjelasan</b>
<i>Login</i>	-	Halaman autentikasi untuk <i>login</i> pengguna dan operator. Proses <i>login</i> menggunakan alamat email dan <i>password</i> yang sudah terdaftar.
Verifikasi OTP	-	Jika email belum terverifikasi maka akan diarahkan ke halaman verifikasi. Verifikasi dengan menggunakan kode OTP yang dikirimkan ke email yang dimasukkan pada saat <i>login</i> .
Lupa <i>password</i>	<i>Login</i>	Pengguna aplikasi dapat memanfaatkan fitur ini jika pengguna kehilangan <i>password</i> dengan cara memasukkan alamat email yang sesuai untuk menerima petunjuk lebih lanjut.
Profil pengguna	<i>Login</i> , Operator	Proses pengaturan pengguna dilakukan pada halaman profil pengguna. Pengguna dapat menyesuaikan informasi diri seperti nama, nomor hp, dan foto profil.
Pengguna	<i>Login</i> , Operator	Halaman pengguna digunakan untuk menampilkan semua pengguna yang terdaftar disertai dengan akses yang dimiliki, serta untuk menambah dan menghapus pengguna.

**Tabel 3.3** Lanjutan.

Nama	Autentikasi	Penjelasan
Penjadwalan	<i>Login, Operator</i>	Halaman penjadwalan digunakan untuk mengatur jadwal seperti menambahkan, memperbarui, dan menghapus jadwal.
Riwayat akses	<i>Login, Operator</i>	Halaman ini digunakan untuk menampilkan semua aktivitas pengguna pada sistem keamanan kunci pintu gedung.

Untuk meningkatkan kenyamanan dan performa dari *website* maka pembuatan setiap tampilan akan menggunakan *livewire*. *Livewire* merupakan sebuah *library* pada laravel yang digunakan untuk menerapkan konsep *Single Page Application* (SPA) pada Laravel. Laravel sendiri merupakan kerangka kerja yang bersifat *server-side* yaitu semua pengolahan data akan dilakukan di dalam *server* dan hasilnya akan dikirimkan kembali ke *client* berupa tampilan dalam format http. Oleh sebab itu, setiap ada interaksi pengguna maka seluruh halaman akan diperbarui sehingga menjadikan tampilan kurang nyaman dan dapat menurunkan performa *website* karena harus memuat seluruh halaman dari awal.



**Gambar 3.13** Cara kerja *livewire*.

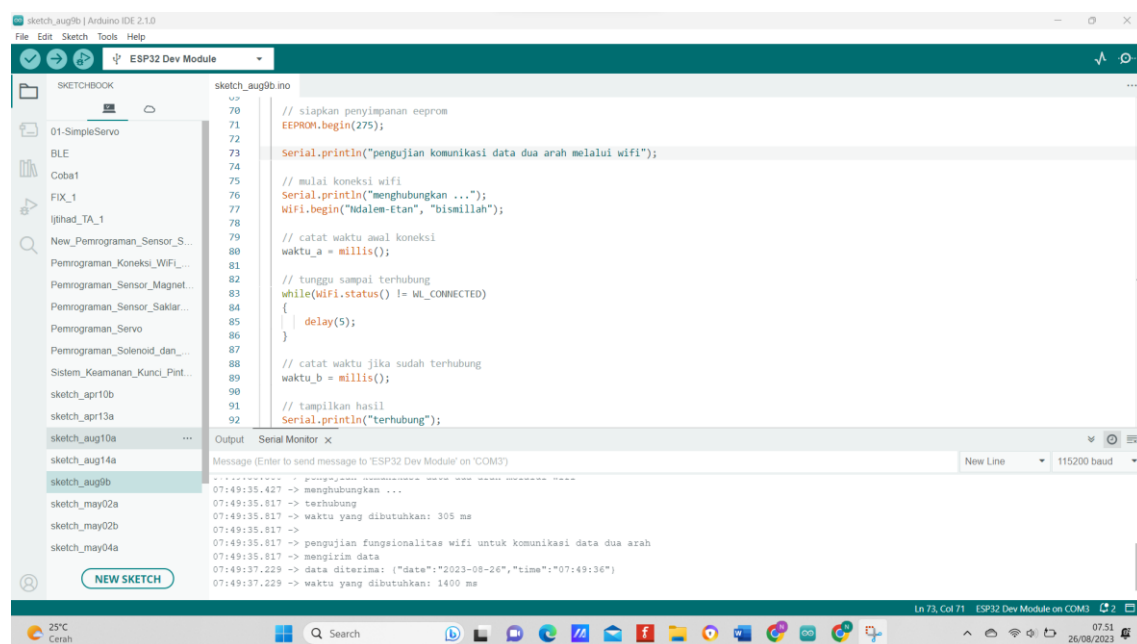
Pada Gambar 3.13 terlihat cara kerja dari *library livewire*. *Livewire* bekerja dengan menggunakan sebuah *request AJAX* untuk memanggil sebuah metode pada kontroler. Jika pada tampilan pengguna melakukan sebuah aksi seperti menekan tombol maka *livewire* akan mengirimkan sebuah permintaan ke *server* dengan menggunakan

AJAX. Permintaan dikirimkan ke sebuah *endpoint* API yang dibuat secara otomatis oleh *livewire* untuk menjalankan *method-method* yang ada pada kontroler. Kontroler akan mengirimkan respon berupa DOM hasil dari perubahan yang dilakukan. Setelah diterima oleh *client* maka DOM tersebut akan dibantingkan dan digadungkan dengan DOM yang sedang tampil dan kemudian di-*render*. Dengan menggunakan metode tersebut maka proses pembaruan tampilan hanya dilakukan pada bagian yang berubah dengan menggunakan manipulasi DOM tanpa memuat seluruh tampilan dari awal.

## 4. VERIFIKASI

### 4.1 Komunikasi WiFi dan Bluetooth

Dalam pengembangan proyek-proyek elektronik menggunakan *platform* seperti Arduino, komunikasi nirkabel adalah hal yang umum dilakukan. Komunikasi WiFi digunakan untuk mentransfer data atau mengendalikan perangkat dari jarak jauh melalui koneksi internet sedangkan komunikasi *bluetooth* lebih cocok untuk jarak pendek.



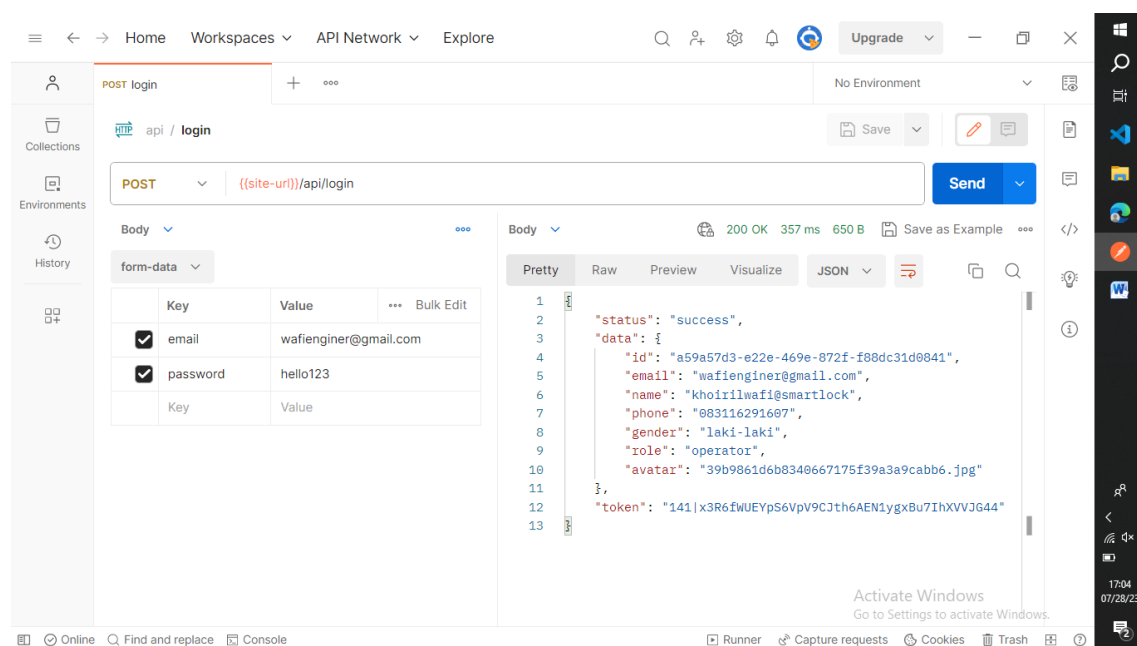
**Gambar 4.1** Tampilan serial monitor arduino.

Gambar 4.1 di atas terlihat tampilan serial monitor Arduino. Serial Monitor adalah alat yang digunakan untuk memantau data yang dikirimkan melalui *port serial* antara mikrokontroler dan komputer. Dapat dilihat pada Gambar 4.1 bahwa komputer berhasil terhubung dan melakukan komunikasi data dengan mikrokontroler melalui koneksi WiFi.

Oleh karena itu, penggunaan Arduino IDE untuk pengembangan dan pemrograman perangkat-perangkat yang menggunakan mikrokontroler dapat digunakan pada proses selanjutnya.

## 4.2 Backend API Laravel

Proses pengembangan *backend* API dilakukan dengan menggunakan kerangka kerja Laravel. Laravel menyediakan modul-modul yang dapat langsung digunakan untuk membuat sebuah API.



**Gambar 4.2** API laravel.

Gambar 4.2 di atas memperlihatkan contoh hasil *request* API dengan menggunakan *postman*. *Request* yang dikirimkan ke *endpoint* “/api/login” merupakan contoh proses *login* melalui API yang akan diimplementasikan pada aplikasi *mobile*. Dapat dilihat pada Gambar 4.2, *postman* berhasil melakukan *login* dengan mendapatkan respon kode 200 dan status sukses sehingga penggunaan laravel untuk membuat sebuah API dapat digunakan pada proses selanjutnya.



### 4.3 Kode QR

Kode QR adalah jenis kode batang yang terdiri dari pola-pola hitam dan putih yang dapat di-*scan* oleh perangkat seperti kamera *smartphone*. Kode QR mengandung informasi tertentu, seperti URL, teks, nomor telepon, atau bahkan informasi kontak. Begitu kode QR di-*scan*, informasi yang terkandung dalam kode dapat dengan mudah diakses oleh perangkat yang melakukan pemindaian.



**Gambar 4.3** Pemindaian kode QR melalui aplikasi *mobile*.

Pada Gambar 4.3 di atas memperlihatkan tampilan layar aplikasi *mobile* saat memproses informasi dari kode QR. Penggunaan kode QR untuk pindai akses cepat akan digunakan pada proses selanjutnya.

## 5. PENUTUP

Dokumen B300 menjelaskan desain perancangan sistem keamanan kunci pintu gedung berbasis IoT yang meliputi perancangan subsistem komunikasi data dua arah, perangkat penguncian, *database*, *server*, metode untuk melakukan verifikasi akses, dan aplikasi *mobile*. Hasil perancangan desain sistem keamanan kunci pintu gedung berbasis IoT akan menjadi acuan dalam proses implementasi.