Pengembangan Sistem Kontrol Pengisian Daya Mobil Listrik Berbasis Aplikasi Web dan IoT

Rafli Rajendra Permana[1], Nur Rohman Rosyid[1*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada
rafli.rajendra0303@mail.ugm.ac.id
[*]Korespondensi: nrohmanr@ugm.ac.id;

Abstract – The adoption of electric vehicles in Indonesia has shown substantial growth, with electric car sales increasing by 39.91% in May 2024 compared to the previous year. This growth has catalyzed the development of supporting infrastructure, including charging stations across major cities. While many charging stations are transitioning to automated systems, some facilities still operate manually, requiring technical assistance for operation after payment, which reduces efficiency compared to self-service systems. This study presents the development of an IoT-based token system with automatic timer functionality for electric vehicle charging stations. The system architecture integrates an ESP32 microcontroller and SCT-013 current sensor for usage detection, coupled with a Laravel framework-based web application for user interface and MQTT protocol for device-application communication. Testing demonstrated successful integration between IoT devices and the web application, with the token system enabling repeated usage within a 24-hour period until the allocated duration is depleted. The automatic timer activates when current detection exceeds 0.5 A and temporarily pauses when current drops below 0.3 A. Real-time monitoring capabilities were achieved through synchronized interfaces between the charging station and monitoring systems, ensuring accurate usage tracking and system status updates.

Keywords – charging station, IoT, Laravel, MQTT, LAMP Stack

Intisari – Penggunaan kendaraan listrik di Indonesia telah mengalami peningkatan signifikan, dengan kenaikan penjualan mobil listrik sebesar 39,91% pada Mei 2024 dibandingkan tahun sebelumnya. Kenaikan ini mendorong berkembangnya infrastruktur pendukung, termasuk *charging station* di kota-kota besar. Sementara banyak *charging station* telah beralih ke sistem otomatis, beberapa fasilitas masih dioperasikan secara manual yang membutuhkan bantuan teknisi setelah pembayaran, sehingga kurang efisien dibandingkan sistem *self-service*. Penelitian ini menyajikan pengembangan sistem *token* berbasis IoT dengan fungsi *timer* otomatis untuk *charging station* kendaraan listrik. Arsitektur sistem mengintegrasikan *microcontroller* ESP32 dan sensor arus SCT-013 untuk deteksi penggunaan, dikombinasikan dengan aplikasi web berbasis Laravel *framework* sebagai antarmuka pengguna dan protokol MQTT untuk komunikasi perangkat-aplikasi. Pengujian menunjukkan keberhasilan integrasi antara perangkat IoT dan aplikasi web, dengan sistem *token* yang memungkinkan penggunaan berulang dalam rentang waktu 24 jam hingga durasi yang dialokasikan habis. *Timer* otomatis aktif ketika deteksi arus melebihi 0,5 A dan berhenti sementara saat arus turun di bawah 0,3 A. Kemampuan pemantauan *real-time* dicapai melalui antarmuka yang tersinkronisasi antara *charging station* dan sistem *monitoring*, memastikan pelacakan penggunaan dan pembaruan status sistem yang akurat.

Kata kunci – stasiun pengisian daya, IoT, Laravel, MQTT, LAMP *Stack*

I. PENDAHULUAN

Selama setahun terakhir, terjadi peningkatan pesat dalam penggunaan kendaraan listrik sebagai kendaraan pribadi. Berdasarkan data Gabungan Industri Kendaraan Bermotor Indonesia, penjualan mobil listrik mencapai 6.033 unit pada Mei 2024, menunjukkan kenaikan sebesar 39,91% dibandingkan dengan Mei 2023. Peningkatan permintaan terhadap teknologi kendaraan dengan sumber energi bersih terjadi sebagai upaya pengurangan emisi karbon. Transisi menuju penggunaan energi yang lebih ramah lingkungan menjadi langkah penting dalam pembangunan berkelanjutan [1].

Peningkatan jumlah mobil listrik seharusnya diiringi dengan penambahan stasiun pengisian daya (*charging station*). Dalam skenario *Business as Usual* (BAU), kebutuhan stasiun pengisian daya di Indonesia diprediksi akan meningkat dari 7.953 unit pada tahun 2025 menjadi 202.424 unit pada tahun 2040. Pembangunan infrastruktur pengisian daya untuk menunjang pertumbuhan jumlah mobil listrik perlu dilakukan dengan tepat, mengikuti standar perancangan dan pembangunan infrastruktur seperti yang telah diterapkan di Amerika Serikat. Dibandingkan dengan SPBU konvensional, stasiun pengisian daya mobil listrik memiliki keunggulan karena lebih mudah ditempatkan di berbagai lokasi umum, termasuk area parkir. Ukuran stasiun pengisian yang relatif kecil dan tidak memerlukan tangki penyimpanan bahan bakar menjadi faktor utama kemudahan penempatan tersebut [2].

Pengoperasian stasiun pengisian memerlukan bantuan teknisi atau *engineer* yang bertugas untuk menyalakan pemutus arus (*circuit breaker*) tiga fase setelah pelanggan melakukan pembayaran di bagian *register*. Sistem manual tersebut memakan waktu karena pihak *register* harus menghubungi *engineer* terlebih dahulu [3]. Pengembangan sistem kontrol stasiun pengisian daya yang dilengkapi mekanisme *token* dan *timer* diusulkan sebagai solusi yang dapat mengotomatisasi proses tersebut [4].

Penelitian terdahulu telah mengembangkan sistem pemantauan dan pencarian lokasi *charging station* berbasis IoT secara *real-time* [5]. Penelitian lain mengembangkan *charging station* universal yang dilengkapi dengan *Battery Management System* (BMS) untuk memantau parameter arus, tegangan, dan suhu baterai [6]. Ada pula penelitian yang mengembangkan *Smart EV-Hub* yang memungkinkan pengguna memilih *port* pengisian, memantau status baterai, dan melakukan pembayaran melalui aplikasi *mobile* berbasis

Android [7]. Namun, penelitian-penelitian tersebut belum mengimplementasikan sistem otomatisasi pengisian daya berbasis *token* dengan *timer* yang dapat berjalan otomatis berdasarkan deteksi arus.

Penelitian ini bertujuan untuk mengembangkan sistem kontrol *charging station* terintegrasi yang dilengkapi dengan mekanisme *token* dan *timer* otomatis untuk menggantikan sistem manual yang ada. Sistem ini menggunakan ESP32 sebagai mikrokontroler, sensor arus SCT-013 untuk mendeteksi penggunaan *charging station*, aplikasi web berbasis Laravel *framework* sebagai antarmuka pengguna, serta protokol MQTT (*Message Queuing Telemetry Transport*) untuk komunikasi antara perangkat IoT (*Internet of Things*) dan aplikasi web [8,9]. Pengembangan sistem ini diharapkan dapat meminimalisir *human-error* dalam pengoperasian, mempersingkat alur transaksi, dan meningkatkan efisiensi operasional infrastruktur pengisian daya [10].

II. DASAR TEORI

A. *Internet of Things* (IoT)

International Telecommunication Union (ITU) mendefinisikan IoT sebagai "sebuah infrastruktur global untuk masyarakat informasi yang memungkinkan layanan canggih dengan menghubungkan benda-benda fisik dan virtual berdasarkan teknologi informasi dan komunikasi yang ada dan berkembang". IoT dapat didefinisikan sebagai jaringan benda-benda fisik yang saling terhubung dan dilengkapi dengan sensor serta perangkat lunak. Jaringan ini memungkinkan benda-benda tersebut mengumpulkan, mengirim, dan memproses data secara mandiri dengan menggunakan komputasi awan sebagai pusat integrasi perangkat [7].

Terdapat tiga arsitektur IoT yang paling umum, yaitu *3-layer Architecture*, *SoA-based Architecture*, dan *Middleware-based Architecture*. *3-layer Architecture* merupakan arsitektur paling *generic* dari ketiga arsitektur tersebut, yang disusun oleh *application layer*, *network layer*, dan *perception layer*. *Perception layer* berperan sebagai tingkat fisik dari objek yang terhubung dan bertugas mengumpulkan informasi dari lingkungan sekitar. Informasi ini diproses dan dikirimkan ke *application layer* melalui *network layer* dalam bentuk data. *Application layer* kemudian memproses data tersebut sesuai kebutuhan [8].

B. Sensor Arus Listrik

Sensor arus terdiri dari dua jenis utama, yaitu sensor arus invasif dan non-invasif. Sensor arus invasif atau sensor arus *shunt* bekerja dengan mengukur tegangan yang jatuh pada resistor *shunt* yang dipasang secara seri dengan beban. Resistor *shunt* memiliki nilai resistansi yang sangat kecil, biasanya dalam orde miliohm, sehingga tidak mengganggu rangkaian secara signifikan [11].

Sensor arus non-invasif bekerja berdasarkan prinsip induksi elektromagnetik. Arus listrik yang mengalir melalui sebuah konduktor akan menghasilkan medan magnet di sekitarnya. Medan magnet ini dideteksi menggunakan transformator arus (*current transformer*) yang terdiri dari inti ferit berbentuk cincin dan lilitan kawat tembaga. Ketika konduktor yang dialiri arus AC dilewatkan melalui inti ferit, medan magnet yang terbentuk menginduksi tegangan pada lilitan sekunder transformator arus [11].

C. *Web Application*

Aplikasi web merupakan perangkat lunak yang berjalan di server dan diakses melalui sebuah *device* menggunakan jaringan internet, umumnya dengan protokol HTTP atau HTTPS. Aplikasi ini memiliki arsitektur tiga lapis, yaitu lapisan presentasi (*front-end*), logika aplikasi (*back-end*), dan basis data. Ketiga lapisan ini bekerja secara sinergis dalam memberikan pengalaman interaktif bagi pengguna tanpa memerlukan instalasi perangkat lunak di perangkat lokal.

D. LAMP *Stack*

LAMP *Stack* merupakan seperangkat perangkat lunak yang terdiri dari Linux sebagai sistem operasi, Apache sebagai *web server*, MySQL sebagai sistem manajemen basis data, dan PHP sebagai bahasa pemrograman server. Linux dipilih karena sifatnya yang *open-source*, ringan, dan memiliki dokumentasi yang mudah diakses. Apache digunakan sebagai *web server* karena konfigurasinya yang sederhana serta kompatibilitasnya dengan berbagai modul. MySQL dan PHP dipilih karena ketersediaan dokumentasi dan pustaka yang luas [12].

E. Laravel *Framework*

Laravel merupakan *framework* berbasis PHP yang menggunakan pola arsitektur *Model-View-Controller* (MVC). Model berinteraksi langsung dengan tabel di basis data, *Controller* menjadi pusat logika bisnis, dan *View* merupakan antarmuka pengguna. Laravel menyediakan berbagai fitur bawaan seperti *routing*, *middleware*, validasi, *caching*, dan autentikasi yang memfasilitasi pengembangan *web* modern [13].

F. MQTT

MQTT (*Message Queuing Telemetry Transport*) adalah protokol transmisi data yang dikembangkan oleh IBM dan Eurotech. Protokol ini banyak digunakan sebagai media komunikasi untuk *Machine to Machine* (M2M), *Wireless Sensor Networks* (WSN), dan IoT [14]. MQTT menggunakan arsitektur *publish/subscribe* atau *pub/sub* dalam penyediaan *messaging service*. Protokol ini memiliki tiga komponen utama: *Publisher* (pengirim *message*), *Broker* (server yang mengelola *message*), dan *Subscriber* (penerima *message*) [15].

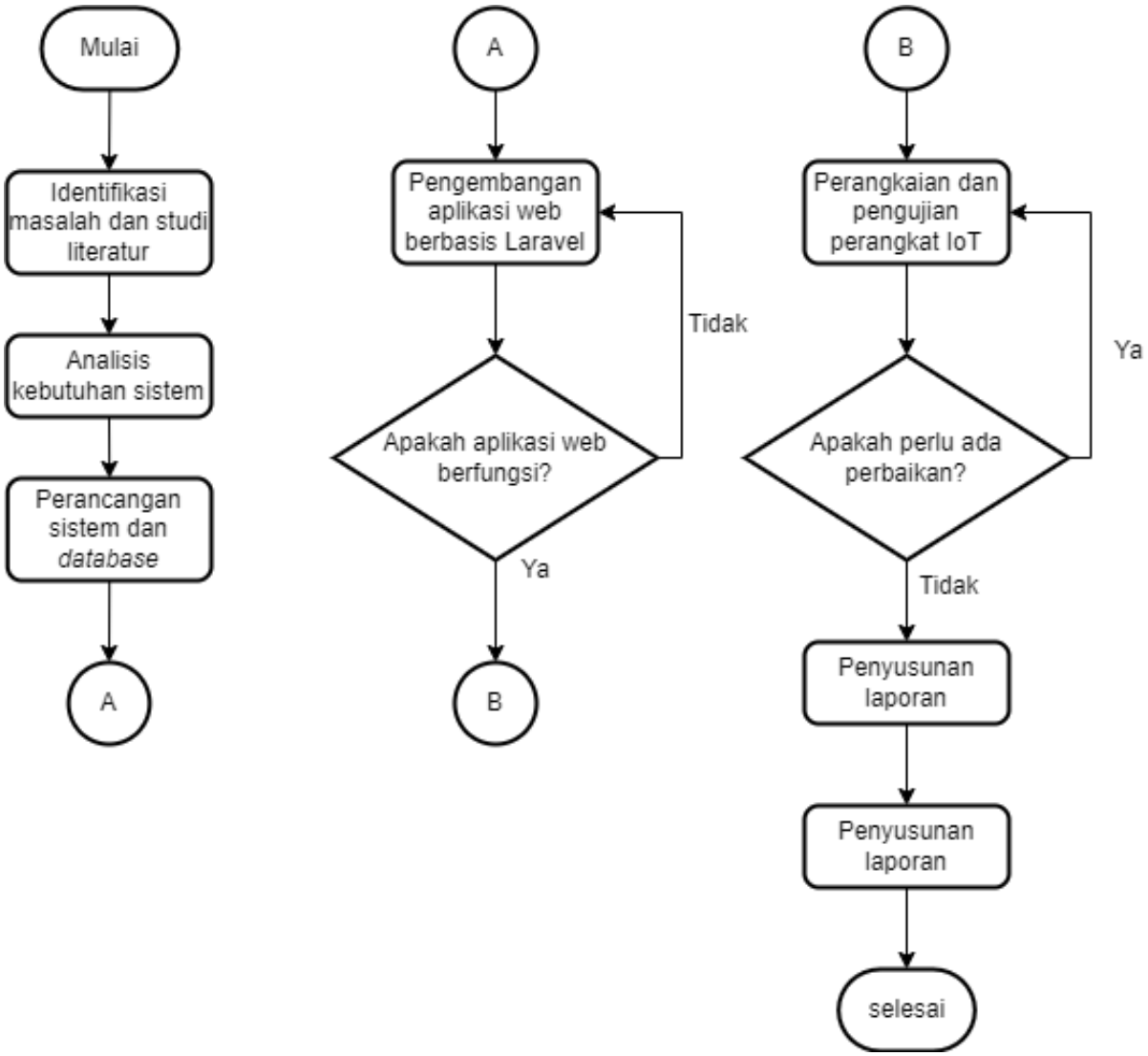
III. METODOLOGI

A. Metodologi Penelitian

Penelitian ini dilakukan dalam beberapa tahap yang saling terkait untuk mencapai tujuan pengembangan sistem kontrol *charging station*. Tahapan pertama adalah analisis kebutuhan sistem melalui observasi langsung terhadap proses pengoperasian *charging station* yang masih manual. Hasil observasi digunakan sebagai dasar perancangan sistem yang terdiri dari perancangan arsitektur, desain basis data, dan desain antarmuka pengguna.

Tahap pengembangan dilakukan secara paralel untuk dua komponen utama sistem. Komponen pertama adalah aplikasi web yang meliputi pembuatan basis data, pemrograman *back-end*, dan pemrograman *front-end*. Komponen kedua adalah perangkat IoT yang mencakup pemrograman ESP32, perancangan rangkaian pada PCB, dan perakitan komponen. Kedua komponen kemudian diintegrasikan melalui MQTT *broker* sebagai media komunikasi.

Tahap terakhir adalah pengujian sistem secara menyeluruh untuk memastikan semua komponen dapat bekerja sesuai dengan rancangan. Alur tahapan penelitian ini diilustrasikan pada Gambar 1.

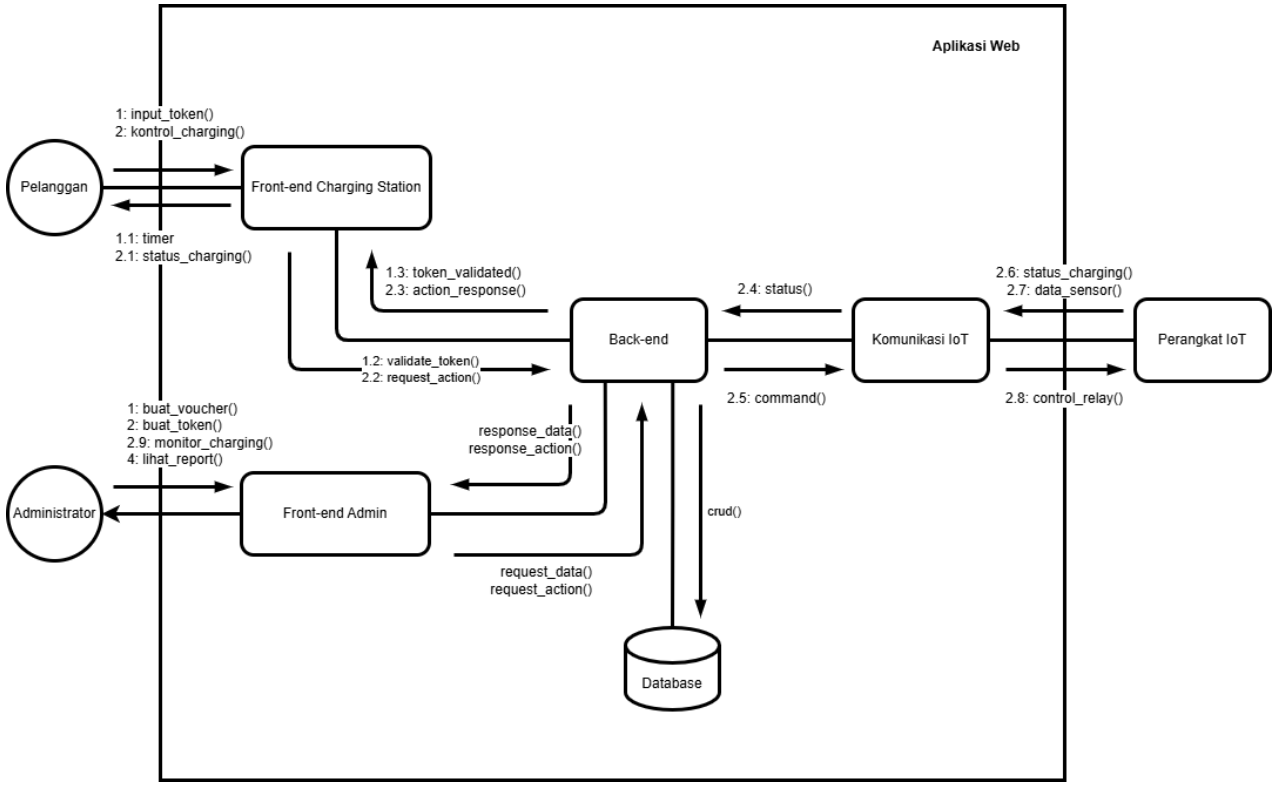


Gambar 1. *Flowchart* Metodologi Penelitian

B. Perancangan Sistem

1. Rancangan Umum Sistem

Sistem yang dikembangkan terdiri dari dua komponen utama: aplikasi web berbasis *full stack* dan perangkat IoT. Aplikasi web berfungsi sebagai antarmuka bagi administrator dalam pembuatan *token* dan pengelolaan urusan administrasi, serta bagi pelanggan dalam memasukkan *token* dan pemantauan durasi *charging* melalui *timer*. Perangkat IoT yang terdiri dari ESP32, sensor arus, dan *relay* berperan mengendalikan *magnetic contactor* dan mengirimkan data ke server berdasarkan pembacaan sensor. Komunikasi antara aplikasi web dan perangkat IoT dilakukan melalui protokol MQTT, seperti yang ditunjukkan pada Gambar 2.



Gambar 2. Diagram Komunikasi Sistem

2. Komponen Perangkat

Perangkat IoT yang dikembangkan menggunakan beberapa komponen utama sebagai berikut:

a. ESP32 DevKit V1

Perangkat IoT menggunakan ESP32 DevKit V1 sebagai mikrokontroler. Spesifikasi ESP32 DevKit V1 tercantum pada Tabel 1.

Tabel 1. Spesifikasi ESP32 DevKit V1

Item	Spesifikasi
Mikrokontroler	*Dual-core* LX6 *microprocessor*
Tegangan Operasi (V)	2,3-3,6
Tegangan Input (V)	4-12
ROM (KB)	448
SRAM (KB)	520
Clock Speed (MHz)	*Up to* 240
Programmable GPIO	30

b. Sensor SCT-013 100

Sensor arus SCT-013 100 merupakan sensor arus AC yang tergolong ke dalam jenis *current transformer*. Sensor ini bersifat non-invasif. Spesifikasi sensor tertera pada Tabel 2.

Tabel 2. Spesifikasi Sensor SCT-013 100

Item	Spesifikasi
Tegangan kerja (V)	660
Arus input terukur (A)	100
Arus *output* terukur (mA)	50
Suhu kerja (C)	-25° ~+70°
Rentang frekuensi	50 Hz – 1 KHz
Tegangan kerja (V)	660
Arus input terukur	100

c. *Power Supply* Hi-Link HLK-5M05

Komponen ini merupakan modul *power supply* berukuran kecil yang berfungsi untuk mengubah listrik AC (220 V) menjadi listrik DC (5 V). Spesifikasi HLK-5M05 tertera pada Tabel 3.

Tabel 3. Spesifikasi Hi-Link HLK-5M05

Item	Spesifikasi
Tegangan input terukur (Vac)	100 – 240
Arus input maksimal (A)	≤0,2
Tegangan *output* terukur (Vdc)	5
Arus *output* maksimal (A)	1

d. Modul *Relay* 5 V 2 *Channel*

Tabel 4 memuat spesifikasi modul *relay* 5V 2 *channel* yang berfungsi sebagai saklar untuk menyalakan dan mematikan *magnetic contactor*.

Tabel 4. Spesifikasi Modul *Relay* 5 V 2 *Channel*

Item	Spesifikasi
Tegangan kerja (Vdc)	5
Tegangan beban (Vac)	125 – 250
Arus beban (A)	10

e. Schneider 3 *Pole Magnetic Contactor*

Komponen ini berperan sebagai "*relay*" untuk listrik 3 fase seperti sebuah modul *relay* yang diaktifkan oleh listrik AC (220 V). Tabel 5 memuat spesifikasi komponen ini.

Tabel 5. Spesifikasi Schneider *Magnetic Contactor*

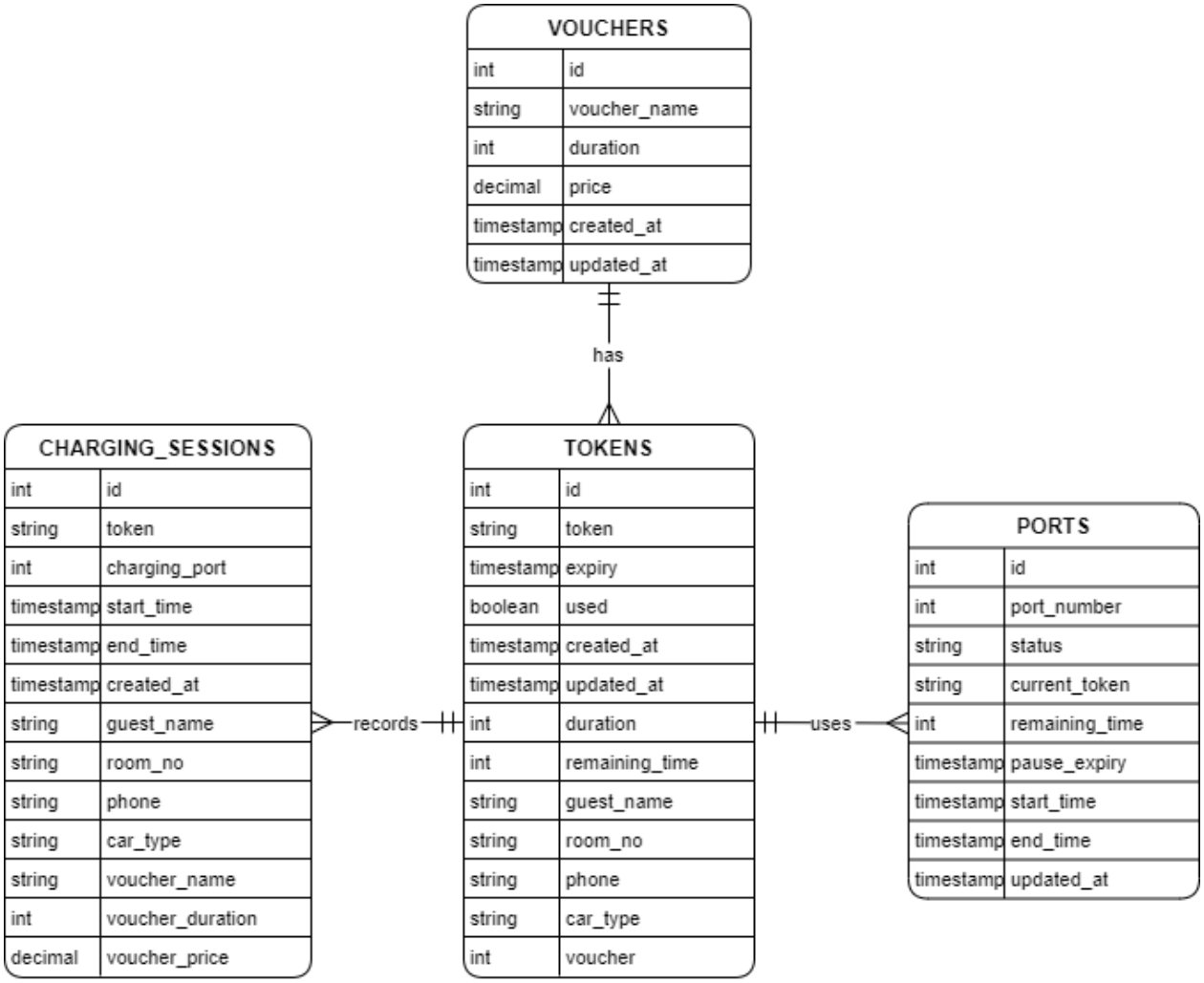
Item	Spesifikasi
Tegangan Operasional	≤ 690 Vac 25-400 Hz (Sirkuit daya), ≤ 300 Vdc (Sirkuit daya), 220 Vac 50/60 Hz (Sirkuit kontrol)
Arus Operasional	60A pada ≤440 Vac (AC-1), 40 A pada ≤440 Vac (AC-3), 40 A pada ≤440 Vac (AC-3e)
Daya Motor	18.5 kW pada 380-400 Vac (AC-3), 11 kW pada 220-230 Vac (AC-3), 22 kW pada 415-440Vac (AC-3), 22 kW pada 500 Vac (AC-3), 30 kW pada 660-690Vac (AC-3)
Arus Termal Konvensional	10 A pada 60°C (sirkuit sinyal), 60 A pada 60°C (sirkuit daya)
Tegangan Kontrol	220V AC 50/60 Hz

3. Perancangan Aplikasi Web

Aplikasi web dirancang menggunakan arsitektur 3-*layer* yang terdiri dari *presentation layer* (*front-end*), *application layer* (*back-end*), dan *data layer*. *Framework* Laravel digunakan dengan pola arsitektur *Model-View-Controller* (MVC) sebagai struktur utama aplikasi.

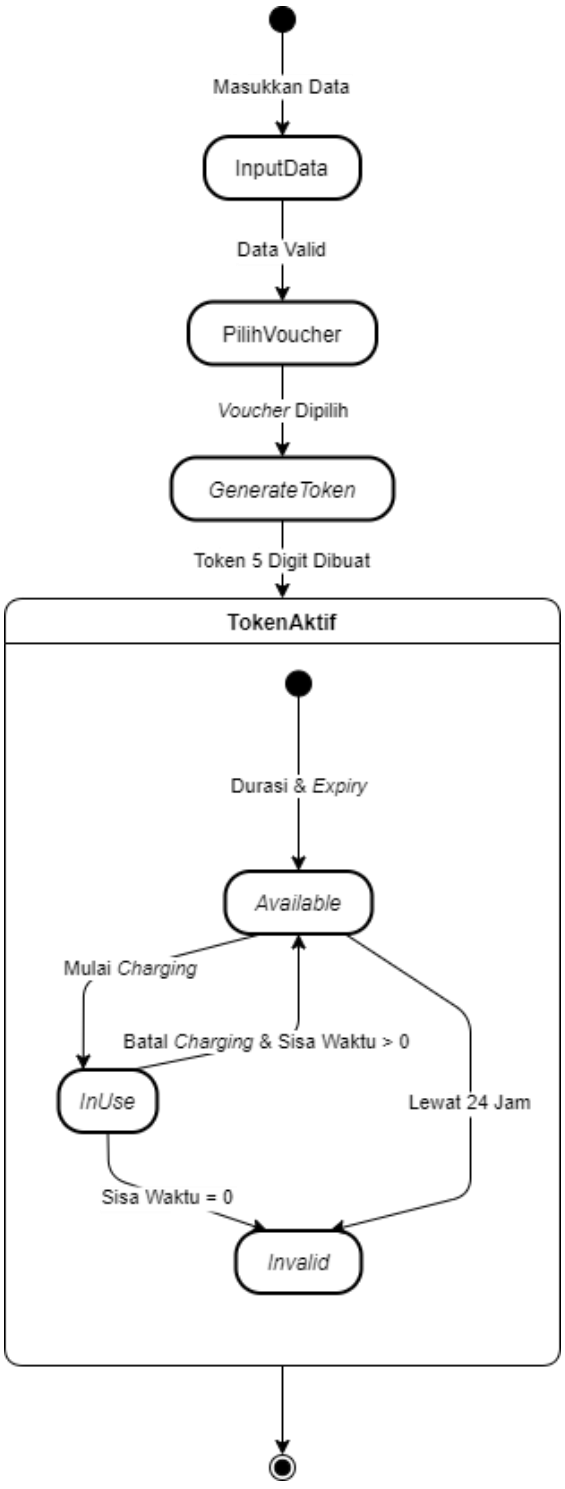
a. Perancangan *Back-end*

Perancangan *back-end* meliputi desain *database* dan penyusunan logika bisnis. Basis data dirancang dengan empat tabel utama yang memiliki relasi seperti ditunjukkan pada Gambar 3.



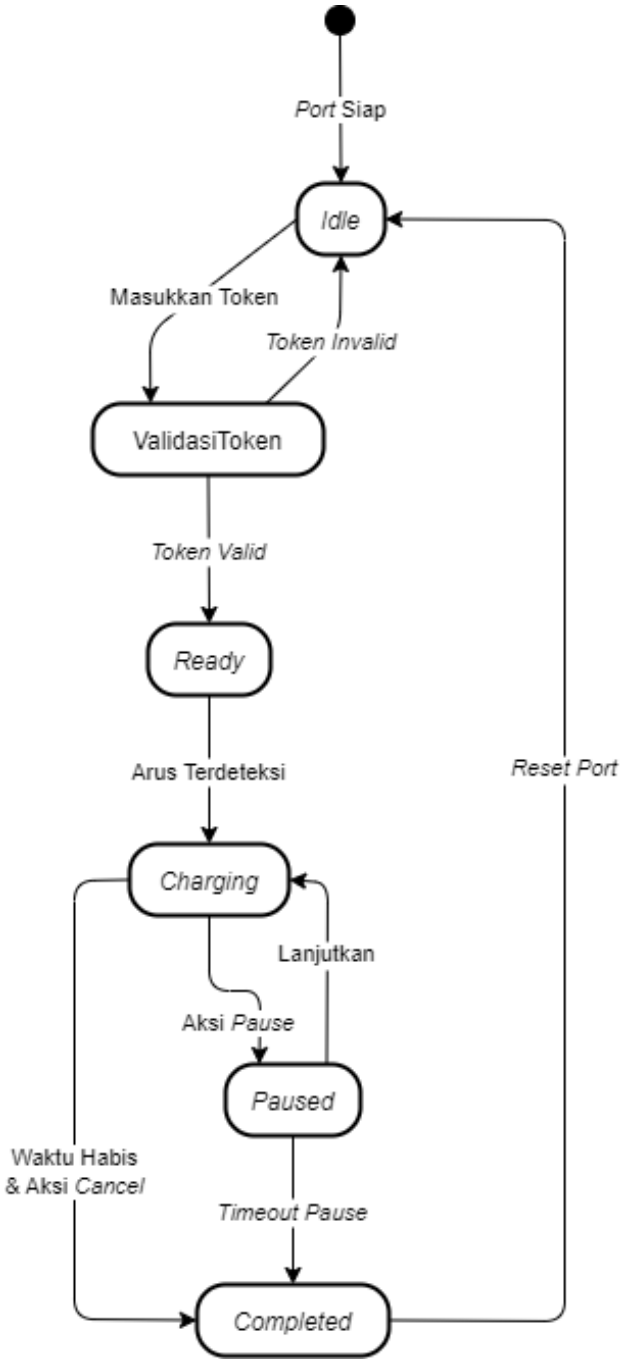
Gambar 3. *ERD Database*

Logika bisnis utama sistem meliputi pengelolaan *token*, *timer* otomatis, dan *monitoring*. Alur kerja sistem *token* ditunjukkan pada Gambar 4.



Gambar 4. *State Diagram Token*

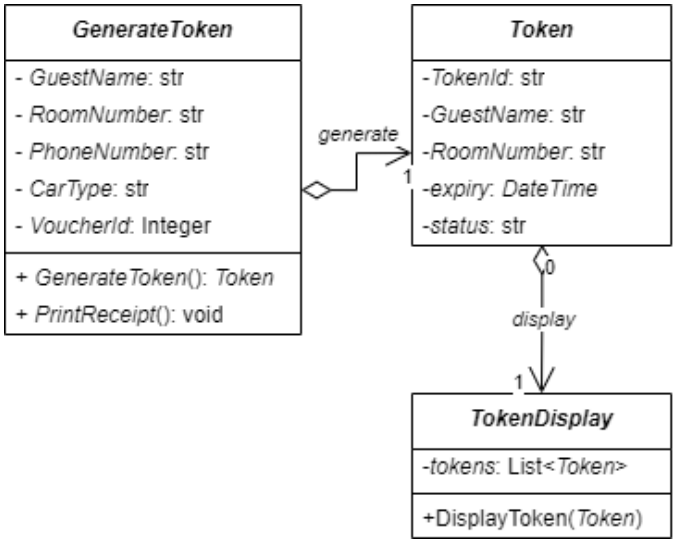
Timer dan kontrol *charging* memiliki alur kerja seperti yang diilustrasikan pada Gambar 5. *Timer* berjalan secara otomatis ketika sistem mendeteksi arus di atas ambang batas yang ditentukan.



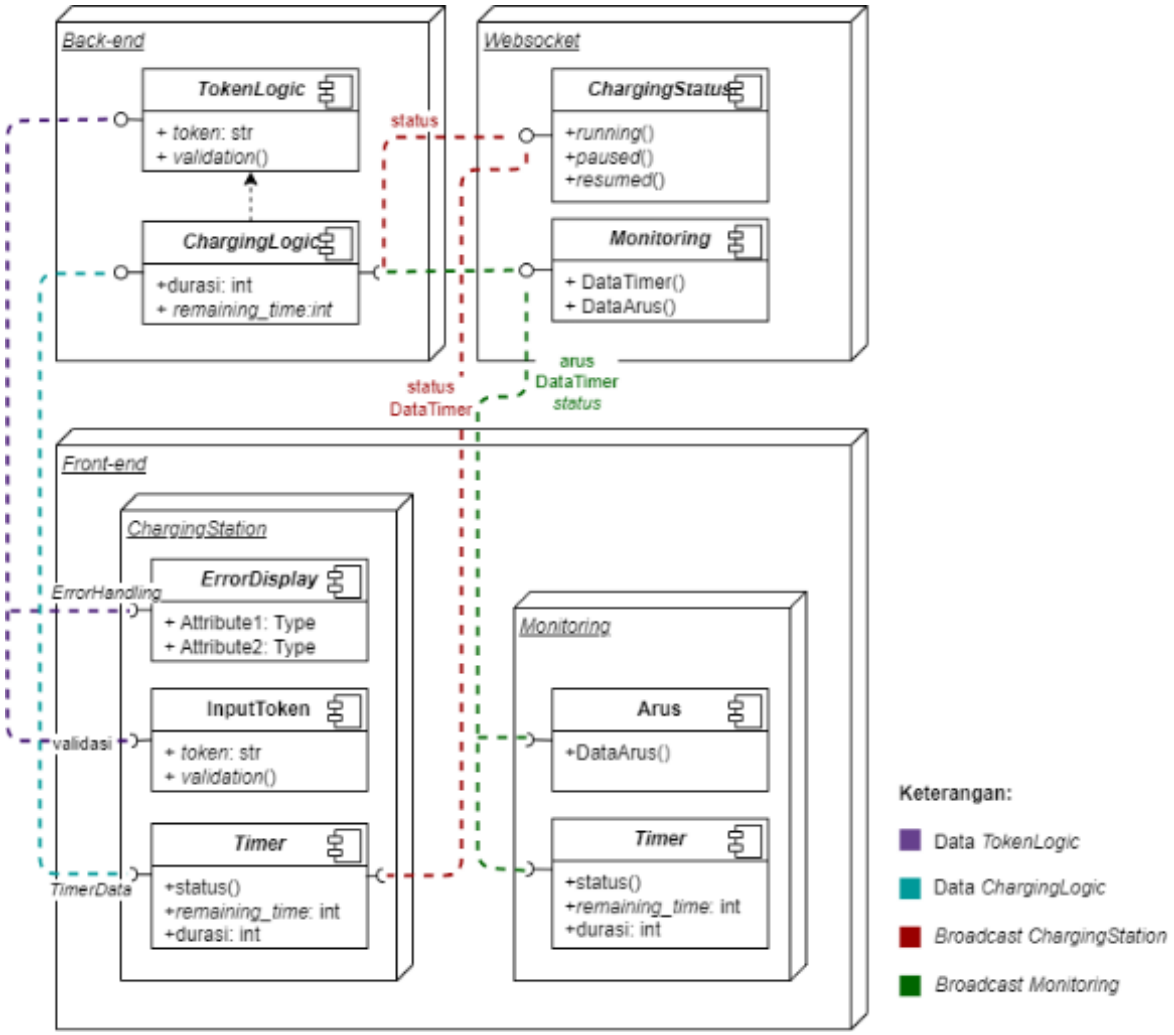
Gambar 5. *State Diagram Timer* dan Kontrol *Charging*

b. Perancangan *Front-end*

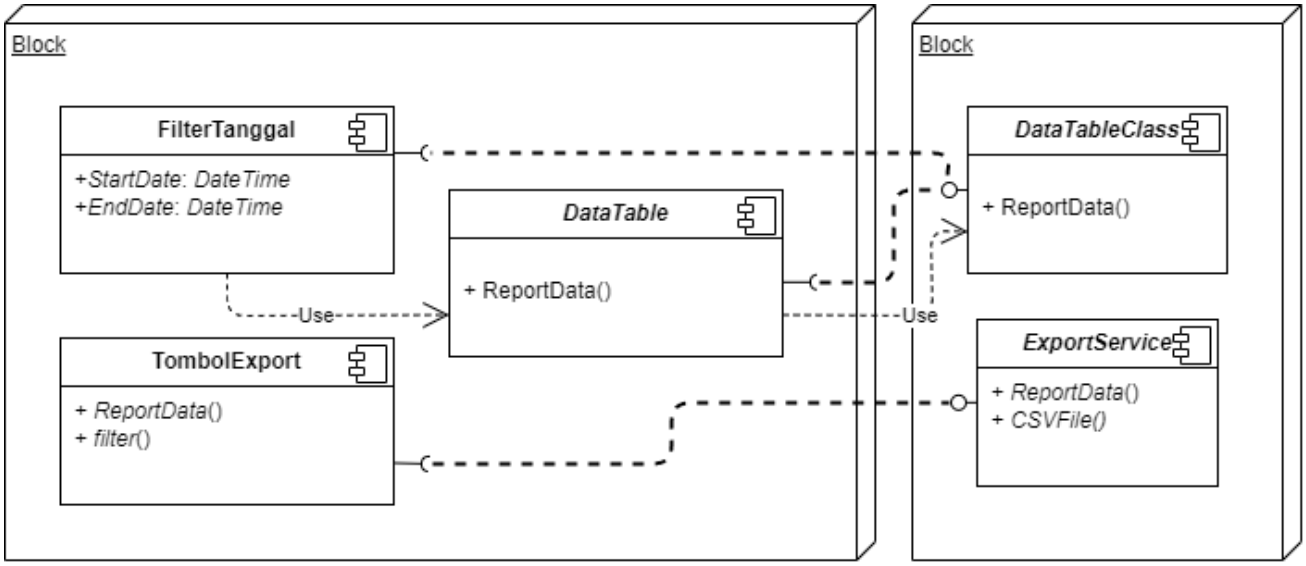
Antarmuka pengguna dikembangkan menggunakan Laravel Blade Template, Alpine.js untuk interaktivitas *client-side*, dan Tailwind CSS untuk *styling*. Sistem memiliki beberapa antarmuka utama dengan struktur komponen yang ditunjukkan pada Gambar 6-8



Gambar 6. *Class Diagram* Antarmuka Pembuatan *Token*



Gambar 7. *Component Diagram* Antarmuka *Charging Station* dan *Monitoring*



Gambar 8. *Component Diagram* Antarmuka *Report*

Sistem memiliki empat antarmuka utama:

- Antarmuka pembuatan *token* untuk administrator dengan *form* pengisian data dan tabel *token*.
- Antarmuka *charging station* untuk pelanggan dengan komponen input *token* dan *timer*.
- Antarmuka *monitoring* untuk administrator dengan tampilan status pengisian dan pembacaan arus secara *real-time*.
- Antarmuka *report* untuk administrator dengan *datatable* yang mendukung *filtering* dan ekspor data.

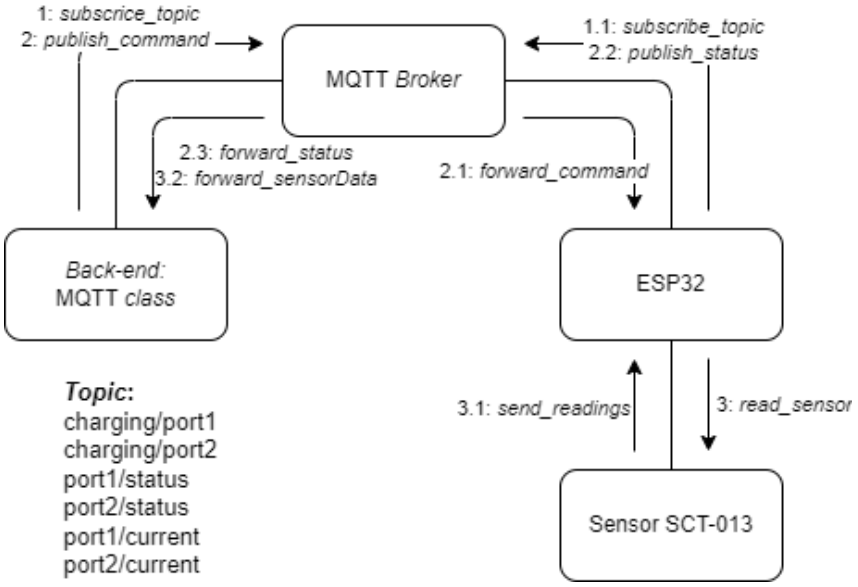
Antarmuka *charging station* dan *monitoring* menggunakan teknologi *websocket* untuk memunculkan *timer* secara otomatis dan *real-time*. *Timer* memiliki dua kondisi utama: "running" saat pengisian daya berlangsung dan "paused" ketika arus turun di bawah ambang batas yang ditentukan.

c. Komunikasi IoT

Komunikasi antara aplikasi web dan perangkat IoT dirancang menggunakan protokol MQTT. Sistem menggunakan tiga topik utama untuk komunikasi:

- "charging/port{1,2}" untuk kontrol pengisian daya
- "charging/port{1,2}/status" untuk status *charging*
- "charging/port{1,2}/current" untuk pembacaan arus

Alur komunikasi IoT ditunjukkan pada Gambar 9.



Gambar 9. *Communication Diagram* Komunikasi IoT

Sistem komunikasi IoT yang dikembangkan menggunakan pola *publish*/*subscribe* melalui MQTT broker. Alur komunikasi terjadi dalam beberapa tahap:

1. *Back-end* Laravel berperan sebagai *publisher* dan *subscriber*:
   - *Subscribe* ke topik status dan *current* untuk menerima pesan dari ESP32.
   - *Publish* perintah ke topik "charging/port{1,2}" untuk mengendalikan ESP32.
2. ESP32 berperan sebagai *publisher* dan *subscriber*:
   - *Subscribe* ke topik "charging/port{1,2}" untuk menerima perintah dari *back-end*.
   - *Publish* status pengisian dan pembacaan sensor ke topik "port{1,2}/status" dan "port{1,2}/current".
3. MQTT *broker* menangani:
   - *Forward* perintah dari *back-end* ke ESP32.
   - *Forward* status dan data sensor dari ESP32 ke *back-end*.
   - Manajemen koneksi dan *reliable message delivery*.

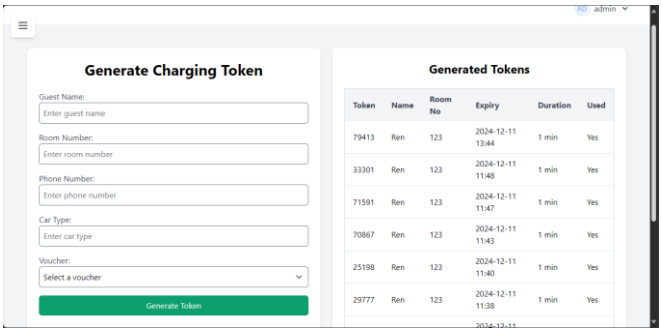
IV. HASIL DAN PEMBAHASAN

A. Hasil Pengembangan Aplikasi Web

Aplikasi web yang dikembangkan berhasil mengimplementasikan seluruh fitur yang direncanakan.

1. *Full Stack* Pembuatan *Token*

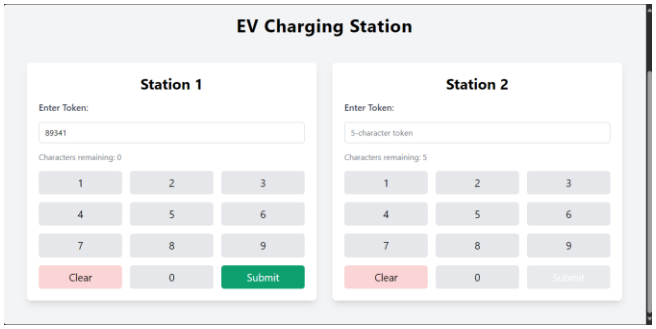
Antarmuka pembuatan *token* menampilkan *form* untuk memasukkan data pelanggan dan *voucher* yang dibeli sesuai durasi yang diinginkan. Di samping *form* terdapat tabel penyajian data yang menampilkan 10 *token* terakhir yang sudah dibuat, seperti ditunjukkan pada Gambar 10.



Gambar 10. Tampilan Antarmuka "Generate Token"

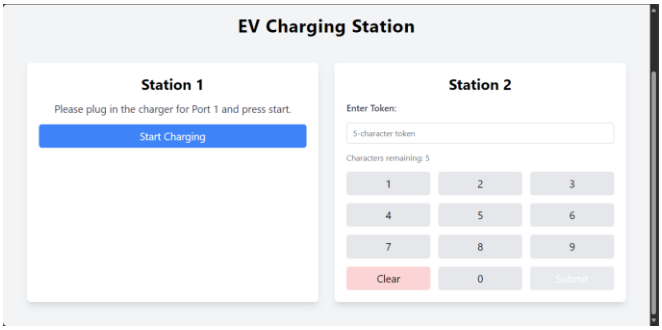
2. *Full Stack Charging Station*

Antarmuka *charging station* menyediakan komponen untuk memasukkan *token* dengan *pad* angka yang dirancang kompatibel dengan monitor *touchscreen*. Gambar 11 menunjukkan tampilan saat pelanggan memasukkan *token*.



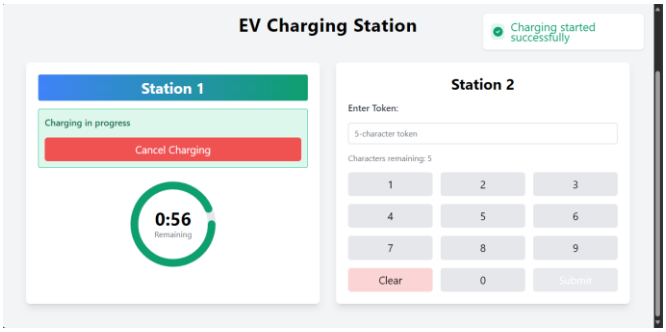
Gambar 11. Tampilan Masukkan *Token*

Setelah validasi *token* berhasil, sistem menampilkan instruksi penggunaan dan tombol "Start Charging" seperti pada Gambar 12.



Gambar 12. Tampilan Setelah Memasukkan *Token*

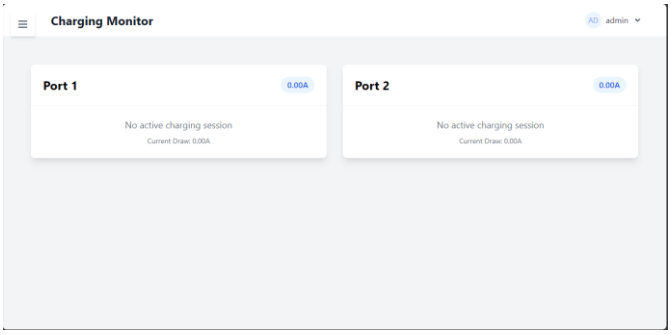
Ketika pengisian daya dimulai, antarmuka menampilkan *timer* yang berjalan dan tombol "Cancel Charging" seperti ditunjukkan pada Gambar 13.



Gambar 13. Tampilan Antarmuka Ketika Sedang *Charging*

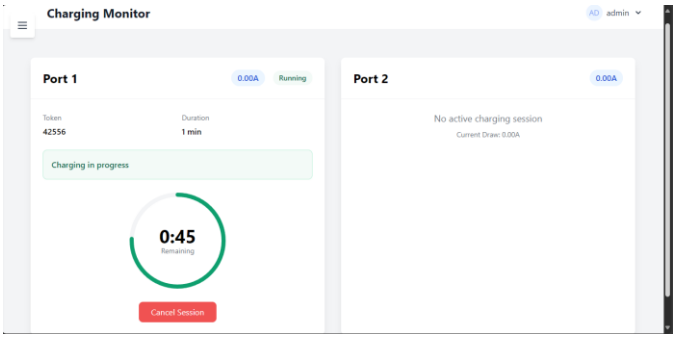
3. Sistem *Monitoring Real-time*

Antarmuka *monitoring* dirancang untuk administrator dalam pemantauan status pengisian daya dan pembacaan arus secara *real-time*. Tampilan awal antarmuka *monitoring* ditunjukkan pada Gambar 14.



Gambar 14. Tampilan Awal Antarmuka *Monitoring*

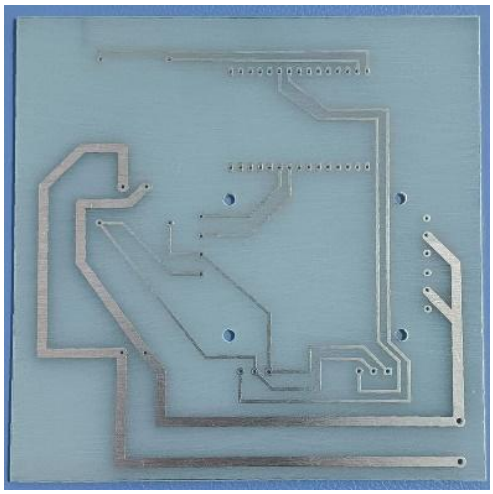
Ketika sesi pengisian daya berlangsung, antarmuka *monitoring* menampilkan status pengisian, durasi tersisa, dan pembacaan arus seperti pada Gambar 15.



Gambar 15. Tampilan Antarmuka *Monitoring* Ketika Sedang Charging

B. Hasil Rancangan Perangkat IoT

Perangkat IoT dikembangkan dalam bentuk PCB (*Printed Circuit Board*) yang menghubungkan komponen-komponen utama sistem. Desain PCB dicetak dengan ukuran 12 cm × 12 cm seperti ditunjukkan pada Gambar 16.



Gambar 16. PCB yang Telah Dicetak

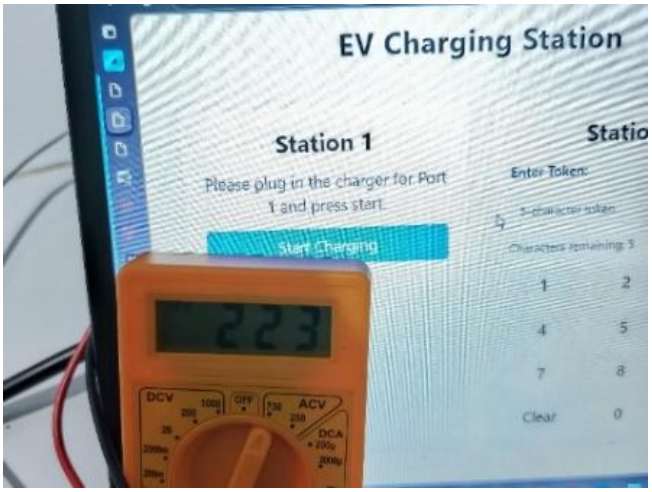
Komponen-komponen yang terpasang pada PCB meliputi ESP32, modul *relay*, sensor arus SCT-013, dan *power supply* HLK-5M05. Hasil pemasangan komponen pada PCB ditunjukkan pada Gambar 17.

Gambar 17. Komponen Terpasang pada PCB

C. Hasil Pengujian Integrasi

1. Pengujian Komunikasi Sistem

Pengujian pertama dilakukan untuk memverifikasi komunikasi antara aplikasi web dan ESP32, khususnya kemampuan *relay* dalam mengendalikan *magnetic contactor*. Hasil pengukuran tegangan AC pada *control terminal* MC saat *token* dimasukkan menunjukkan nilai 223 V, membuktikan ESP32 berhasil memproses pesan "start" dan menutup *relay* seperti ditunjukkan pada Gambar 18.



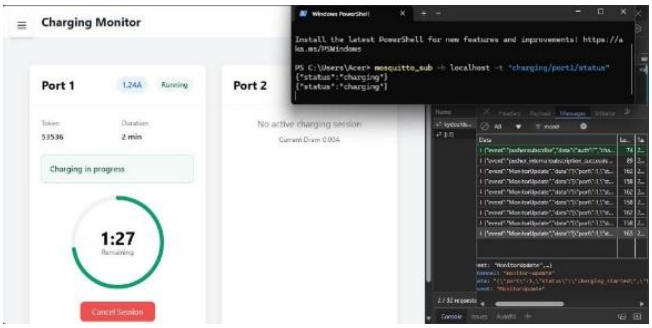
Gambar 18. Pengukuran Tegangan *Control Terminal* MC saat *Token* Dimasukkan

2. Pengujian *Timer* Otomatis

Pengujian dilakukan menggunakan beban setrika untuk mensimulasikan penggunaan *charging station*. Hasil pengujian menunjukkan:

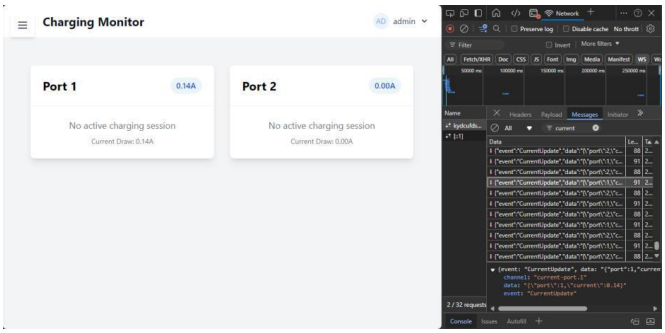
- *Timer* berjalan otomatis ketika arus terdeteksi melebihi 0,5 A.
- Status "paused" aktif saat arus turun di bawah 0,3 A.
- Status "resumed" ketika arus kembali naik di atas 0,5 A.

Gambar 19 menunjukkan tampilan antarmuka *monitoring* saat pengujian dengan beban setrika.



Gambar 19. Tampilan *Timer* yang Berjalan ketika Arus yang Terbaca Melebihi 0,5 A

Pengujian dengan beban kecil menggunakan *smartphone* memverifikasi bahwa *timer* tidak akan berjalan jika arus yang terdeteksi kurang dari 0,3 A, seperti ditunjukkan pada Gambar 20.



Gambar 20. Tampilan *Timer* yang Tidak Berjalan ketika Arus yang Terbaca Kurang dari 0,3 A

3. Akurasi Pembacaan Sensor

Pengujian akurasi sensor SCT-013 100 dilakukan dengan membandingkan pembacaan sensor terhadap alat ukur listrik standar. Hasil pengujian menunjukkan rata-rata selisih pembacaan sebesar 0,17 A, yang dapat diterima mengingat fungsi sensor hanya untuk deteksi penggunaan, bukan pengukuran presisi. Tabel 6 menunjukkan hasil pengujian sensor.

Tabel 6. Hasil Pengujian Sensor

Waktu	Alat Ukur (A)	Sensor SCT-013 100 (A)	Selisih (A)
18:46:21	1,54	1,36	0,18
18:46:22	1,54	1,39	0,15
18:46:23	1,54	1,40	0,14
18:46:24	1,54	1,35	0,19
18:46:25	1,54	1,34	0,2
18:46:26	1,54	1,37	0,17
18:46:27	1,54	1,39	0,15
18:46:28	1,53	1,36	0,18
18:46:29	1,54	1,34	0,2
18:46:30	1,54	1,38	0,16
Rata-rata			0,17

D. Evaluasi Sistem

Hasil pengujian menunjukkan sistem berhasil mengotomatisasi proses pengisian daya dengan fitur-fitur berikut:

- Mekanisme *token* yang memungkinkan penggunaan berulang dalam rentang waktu 24 jam.
- *Timer* otomatis berdasarkan deteksi arus.
- Sistem *pause* dan *resume* otomatis untuk mengakomodasi fluktuasi arus.
- Pemantauan *real-time* melalui antarmuka yang tersinkronisasi.

Sistem ini belum mengantisipasi jika terjadi kegagalan catu daya pada salah satu komponennya. Kegagalan catu daya pada *web server* akan menghentikan *timer* sementara sesi *charging* tetap berjalan, sedangkan kegagalan pada *charging station* menyebabkan *timer* di *web server* tetap berjalan karena perangkat IoT tidak dapat mengirimkan pesan "paused". Kedua situasi ini tidak akan terjadi jika kedua komponen mengalami kegagalan catu daya secara bersamaan.

II. SIMPULAN

Berdasarkan hasil pengembangan dan pengujian sistem kontrol stasiun pengisian daya kendaraan listrik berbasis aplikasi web dan IoT, dapat disimpulkan bahwa sistem yang dikembangkan berhasil mengintegrasikan perangkat IoT dengan aplikasi web melalui protokol MQTT sehingga mampu menggantikan proses manual yang sebelumnya memerlukan bantuan teknisi. Mekanisme *token* yang diimplementasikan memungkinkan penggunaan berulang dalam rentang waktu dua puluh empat jam hingga durasi habis, memberikan fleksibilitas lebih bagi pengguna. Selain itu, *timer* otomatis bekerja sesuai dengan rancangan, yakni aktif ketika arus melebihi 0,5 *ampere* dan berhenti sementara saat arus turun di bawah 0,3 *ampere*. Antarmuka *monitoring* juga berfungsi dengan baik dalam menampilkan status

pengisian daya serta pembacaan arus secara *real-time*, sehingga memudahkan administrator dalam memantau proses pengisian dan meningkatkan efisiensi operasional sistem pengisian daya.

REFERENSI

[1] Nuhansa Mikrefin Yoedo Putra, "Penjualan Mobil Listrik Mei 2024 Naik 39,91%, Chery Omoda E5 Gusur Wuling Cloud EV." Accessed: Sep. 16, 2024. [Online]. Available: https://otomotif.bisnis.com/read/20240614/46/1774197/penjualan-mobil-listrik-mei-2024-naik-3991-chery-omoda-e5-gusur-wuling-cloud-ev

[2] E. Djubaedah, Riza, A. Kurniasari, S. N. E. Eny, and A. Nurrohim, "Projection of the Demand for Charging Stations for Electric Passenger Cars in Indonesia," *Evergreen*, vol. 10, no. 3, 2023, doi: 10.5109/7151723.

[3] K. B. Wicaksono and A. Aprianingsih, "Electric Car Penetration Potential in Indonesia," *Jurnal Penelitian Transportasi Darat*, vol. 23, no. 2, pp. 142–149, Dec. 2021, doi: 10.25104/jptd.v23i2.1803.

[4] E. A. Mohammed, M. H. Qahtan, and A. J. Ali, "Internet of things based real-time electric vehicle and charging stations monitoring system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 27, no. 3, pp. 1661–1669, Sep. 2022, doi: 10.11591/ijeecs.v27.i3.pp1661-1669.

[5] N. Bhuvaneswary, M. V. Karthik, N. P. Kumar, P. M. Kumar, and P. S. Harsha, "Universal Charging Station for EV and BMS," in *7th International Conference on Trends in Electronics and Informatics, ICOEI 2023 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 135–140. doi: 10.1109/ICOEI56765.2023.10125949.

[6] R. S. Vijayashanthi, S. Prithiviraj, S. Prathish, M. P. Nirmal, J. Mohan Kumar, and R. Manoj Kumar, "Smart EV-Hub: A Smart Development of an Internet of Things based Electric Vehicle Charging Station using Artificial Intelligence," in *Proceedings - 3rd International Conference on Advances in Computing, Communication and Applied Informatics, ACCAI 2024*, Institute of Electrical and Electronics Engineers Inc., 2024. doi: 10.1109/ACCAI61061.2024.10602038.

[7] I. Zhou *et al.*, "Internet of Things 2.0: Concepts, Applications, and Future Directions," *IEEE Access*, vol. 9, pp. 70961–71012, 2021, doi: 10.1109/ACCESS.2021.3078549.

[8] M. Lombardi, F. Pascale, and D. Santaniello, "Internet of things: A general overview between architectures, protocols and applications," *Information (Switzerland)*, vol. 12, no. 2, pp. 1–21, Feb. 2021, doi: 10.3390/info12020087.

[9] B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *IEEE Access*, vol. 8, pp. 201071–201086, 2020, doi: 10.1109/ACCESS.2020.3035849.

[10] G. Rajendran, C. A. Vaithilingam, N. Misron, K. Naidu, and M. R. Ahmed, "A comprehensive review on system architecture and international standards for electric vehicle charging stations," *J Energy Storage*, vol. 42, p. 103099, Oct. 2021, doi: 10.1016/J.EST.2021.103099.

[11] S. Ziegler, H. H. C. Iu, R. C. Woodward, and L. J. Borle, "Theoretical and practical analysis of a current sensing principle that exploits the resistance of the copper trace," in *2008 IEEE Power Electronics Specialists Conference*, 2008, pp. 4790–4796. doi: 10.1109/PESC.2008.4592730.

[12] Vladimir Kaplarevic, "What Is LAMP?" Accessed: Jan. 09, 2025. [Online]. Available: https://phoenixnap.com/kb/what-is-a-lamp-stack#ftoc-heading-10

[13] O. Widodo Purbo, "A Systematic Analysis: Website Development using Codeigniter and Laravel Framework," 2021.

[14] Y. Sasaki and T. Yokotani, "Performance evaluation of MQTT as a communication protocol for IoT and prototyping," *Advances in Technology Innovation*, vol. 4, no. 1, pp. 21–29, Jan. 2019.

[15] B. Mishra and A. Kertész, "The use of MQTT in M2M and IoT systems: A survey," *IEEE Access*, vol. 8, pp. 201071–201086, 2020.