SISTEM MONITORING PENGGUNAAN KENDARAAN KURIR PAKET BERBASIS MIKROKONTROLER DENGAN PENCATATAN JARAK DAN WAKTU TEMPUH HARIAN DENGAN NOTIFIKASI SUARA

Muhammad[1], Dedi Ary Prasetya,S.T,.M.Eng[2]

[1]Program Studi Teknik Elektro, Universitas Muhammadiyah Surakarta

Abstrak

Perkembangan industri e-commerce mendorong peningkatan volume pengiriman paket oleh perusahaan jasa kurir. Namun banyak perusahaan ekspedisi skala kecil dan menengah belum memiliki sistem pemantauan kendaraan yang memadai. Penelitian ini merancang dan mengimplementasikan sistem monitoring kendaraan kurir berbasis mikrokontroler ESP32 yang mengintegrasikan modul GPS Neo-6M, RTC DS3231, sensor DHT22, layar OLED SSD1306, modul SD Card, buzzer, Firebase Realtime Database sebagai penyimpanan awan (cloud storage), dan Telegram Bot API sebagai platform IoT. Sistem menggunakan arsitektur dual-core FreeRTOS agar operasional sensor, transmisi data ke Firebase, dan komunikasi Telegram berjalan paralel tanpa saling mengganggu. Deteksi titik berhenti dilakukan berbasis radius 20 meter selama 30 detik yang lebih andal dibandingkan deteksi berbasis kecepatan pada kondisi sinyal GPS tidak stabil. Data persisten disimpan di SD Card sehingga tidak hilang saat alat dimatikan. Pengujian dilakukan dengan membawa alat pada kendaraan dan dengan memantau fungsional sistem secara berkala di setiap pemberhentian. Hasil pengujian point-to-point pada 6 rute menunjukkan rata-rata error jarak sebesar 4,24% terhadap referensi. Pengujian multi-point berhasil mencatat 11 titik berhenti dengan akurasi jarak kumulatif 0,13% (jarak alat 22,45 km vs referensi 22,42 km pada 9 titik pertama). Seluruh pengujian pencatatan suhu dan waktu menghasilkan data valid (100%). Firebase dan sistem Telegram Bot berhasil menyinkronkan data riil, mengirimkan notifikasi otomatis, dan merespons perintah interaktif dari pengguna secara real-time.

Kata Kunci: ESP32, GPS Neo-6M, Haversine, monitoring kendaraan, Telegram Bot, FreeRTOS, titik berhenti, Firebase Realtime Database.

Abstract

The growth of the e-commerce industry has significantly increased parcel delivery volumes by courier service companies. However, many small and medium-scale expedition companies lack adequate vehicle monitoring systems. This study designs and implements a courier vehicle monitoring system based on the ESP32 microcontroller integrating GPS Neo-6M, RTC DS3231, DHT22 sensor, SSD1306 OLED display, SD Card module, buzzer, Firebase Realtime Database as cloud storage, and Telegram Bot API as the IoT platform. The system employs a FreeRTOS dual-core architecture so sensor operations, Firebase data transmission, and Telegram communications run in parallel without interference. Stop-point detection is performed using a 20-meter radius for 30 seconds, which is more reliable than speed-based detection when GPS signals are unstable. Persistent data is stored on the SD Card so it is retained when the device is powered off. Testing was conducted by carrying the device on the vehicle and periodically monitoring the system's functionality at each stop. Point-to-point tests on 6 routes show an average distance error of 4.24%. Multi-point testing successfully recorded 11 stop points with

cumulative distance accuracy of 0.13% (device distance 22.45 km vs. reference 22.42 km at the first 9 points). All temperature and time recording tests produced valid data (100%). Firebase and the Telegram Bot system successfully synchronized real-time data, delivered automatic notifications, and responded to interactive commands in real-time.

Keywords: ESP32, GPS Neo-6M, Haversine, vehicle monitoring, Telegram Bot, FreeRTOS, stop point, Firebase Realtime Database.

1. PENDAHULUAN

Perkembangan industri logistik pengiriman paket menuntut efisiensi tinggi dalam pemantauan armada. Banyak penyedia jasa penyewaan atau ekspedisi membutuhkan sistem yang dapat memantau pergerakan kendaraan secara *real-time* untuk memberikan rasa aman dan mencegah penyalahgunaan armada (Pardede, Farisi and Arwani, 2017). Pemantauan pergerakan titik lokasi kendaraan (mobilitas) dari satu tempat ke tempat lainnya secara berkala menjadi kunci dalam efektivitas operasional (Rohman and Amalia, 2025). Guna mengatasi masalah tersebut, pengembangan perangkat pelacak berbasis *Internet of Things* (IoT) menggunakan modul *Global Positioning System* (GPS) telah banyak diimplementasikan (Suppa and Wahyuni, 2024).

Dalam praktiknya, modul receiver satelit seperti GPS Neo-6M sangat ideal digunakan pada sistem pelacakan luar ruangan (*outdoor*) karena kemampuannya dalam memetakan posisi kendaraan secara *real-time* sekaligus membatasi radius pergerakan armada (Tafrikhatin, Wulandari and Fauzi, 2021) .Koordinat lokasi kendaraan (*latitude* dan *longitude*) yang diperoleh dari satelit tersebut dapat diproses lebih lanjut menggunakan pangkalan data riil (*real-time database*) seperti Firebase agar tersinkronisasi langsung ke sistem aplikasi pusat(Prasetyo, Himawan and Sihombing, 2026).

Komponen utama yang bertindak sebagai pemroses pada alat pelacak ini adalah mikrokontroler seri ESP32, yang terbukti efisien karena telah dilengkapi konektivitas WiFi *built-in* dengan kemampuan komputasi memori ganda (*dual-core*) (S.Samsugi, 2024). Melalui ESP32, integrasi komunikasi IoT dapat dihubungkan secara nirkabel kepada pengguna jarak jauh menggunakan aplikasi instan pihak ketiga, seperti *Telegram Bot*, yang mampu memberikan laporan pergerakan lokasi tanpa perlu membangun aplikasi khusus (Tsaqib *et al.*, 2025). Notifikasi Telegram ini juga dapat secara otomatis mengirimkan peringatan kepada pihak manajemen ekspedisi apabila sensor mendeteksi ambang batas anomali pada kendaraan (Adiyanto, Rintyarna and Ariyani, 2026)**.**

2. METODE

Dalam menghitung akumulasi jarak antara dua titik koordinat bumi yang dihasilkan secara berkala oleh modul GPS Neo-6M, akurasi perhitungan matematis pada mikrokontroler memegang

peranan penting. Algoritma navigasi yang digunakan untuk mengukur jarak pada permukaan kelengkungan bumi dengan tingkat presisi tinggi adalah Formula *Haversine*, yang kerap diimplementasikan dalam skenario pengantaran barang/kurir dinamis (Fariza, Hasanuddin and Umar, 2022).

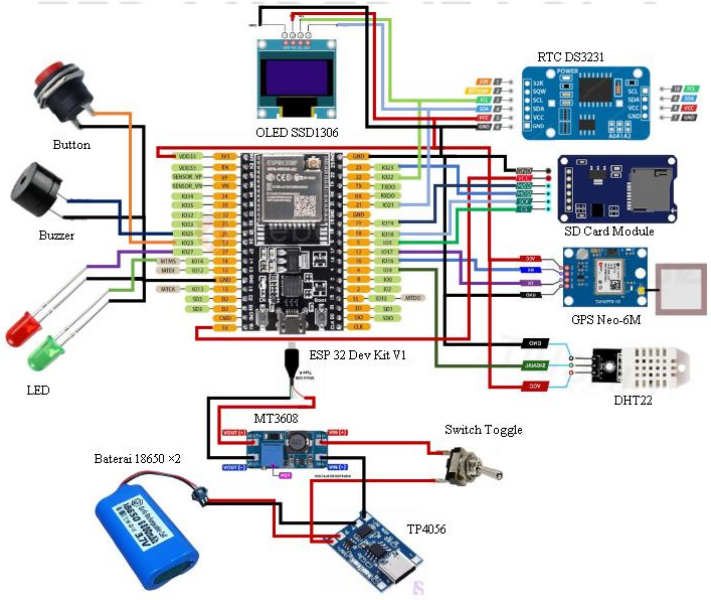
Implementasi algoritma Haversine pada sistem pelacakan dinilai sangat efisien dan tidak memberatkan komputasi pada *board* mikrokontroler karena rumus ini mengubah secara langsung variabel lintang (latitude) dan bujur (longitude) menjadi jarak riil dalam satuan meter atau kilometer (Mau'idzoh Hasanah, Carudin, 2023)

2.1 Perancangan Hardware

Sistem menggunakan ESP32 DevKit V1 38-pin sebagai unit pemroses utama dengan WiFi terintegrasi. Sumber daya menggunakan 2 baterai Li-Ion 18650 3,7V dikonfigurasi paralel (kapasitas ±7000mAh, estimasi runtime ±22 jam), modul charging TP4056 dengan proteksi DW01, dan boost converter MT3608 mengubah 3,7V menjadi 5V yang dimasukkan ke port USB ESP32.

Tabel 1. Komponen hardware sistem monitoring

No	Komponen	Spesifikasi	Fungsi
1	ESP32 DevKit V1	38-pin, 240 MHz, WiFi/BT	Pemroses utama + IoT
2	GPS Neo-6M	Akurasi 2,5 m CEP, UART	Posisi, kecepatan, waktu GPS
3	RTC DS3231	±2 ppm, I2C	Pewaktu presisi harian
4	DHT22	±0,5°C / ±2% RH	Suhu dan kelembaban udara
5	OLED SSD1306	0,96 inci 128×64 px, I2C	Tampilan data real-time
6	SD Card Module	FAT32, SPI	Log data CSV harian
7	Buzzer aktif 5V	Piezoelectric	Notifikasi suara threshold
8	Baterai 18650 ×2	3,7V / 3500mAh paralel	Sumber daya ±22 jam
9	TP4056 + DW01	Max 1A, proteksi DW01	Modul charging baterai
10	MT3608	Step-up 3,7V → 5V	Boost converter ke ESP32



Gambar 2.1. Diagram wiring

Pada Gambar 2.1 memperlihatkan diagram skematik perkabelan (wiring) yang menghubungkan mikrokontroler ESP32 DevKit V1 sebagai pusat pemroses dengan berbagai periferal, meliputi modul GPS, RTC, layar OLED, modul kartu SD, sensor DHT22, dan sistem suplai daya baterai, yang mana wiring bisa dilihat pada table berikut.

`Tabel 2. Tabel Wiring

Modul	Pin Modul	Pin ESP32	Protokol
GPS Neo-6M	TX	16 (RX2)	UART
GPS Neo-6M	RX	17 (TX2)	UART
RTC DS3231	SDA	21	I2C
RTC DS3231	SCL	22	I2C
OLED SSD1306	SDA	21	I2C
OLED SSD1306	SCL	22	I2C
SD Card	MOSI	23	SPI
SD Card	MISO	19	SPI
SD Card	SCK	18	SPI
SD Card	CS	5	SPI
Buzzer	Positive	25	Digital OUT
Tombol reset	IN	26	Digital IN
LED merah	Positive	27	Digital OUT
LED hijau	Positive	14	Digital OUT

Berikut merupakan detail konfigurasi wiring antara mikrokontroler ESP32 DevKit V1 dengan seluruh komponen periferal pendukung sistem. Konfigurasi ini dibagi menjadi tiga aspek koneksi utama:

1. **Jalur Komunikasi Serial UART:** Menghubungkan modul GPS Neo-6M ke Pin RX2 (GPIO 16) dan TX2 (GPIO 17) pada ESP32 untuk pengiriman data koordinat secara *asynchronous*.
2. **Jalur Antarmuka I2C (Inter-Integrated Circuit):** Menghubungkan modul RTC DS3231 (penyedia waktu riil) dan Layar OLED (penampil informasi) secara paralel menggunakan jalur data SDA (GPIO 21) dan jalur jam SCL (GPIO 22).
3. **Jalur Komunikasi SPI (Serial Peripheral Interface):** Digunakan khusus oleh Micro SD Card Shield untuk proses penyimpanan data (*data logging*) massal, yang

memanfaatkan pin standar VSPI pada ESP32 meliputi MOSI (GPIO 23), MISO (GPIO 19), SCK (GPIO 18), dan CS (GPIO 5).

Selain itu, sensor suhu DHT22 dialokasikan pada pin digital tunggal (GPIO 4), serta periferal *output* peringatan suara (*Buzzer*) dikendalikan melalui pin GPIO 12. Sistem pencatatan ini memastikan seluruh pin bekerja tanpa adanya konflik interupsi (*hardware conflict*) pada mikrokontroler.

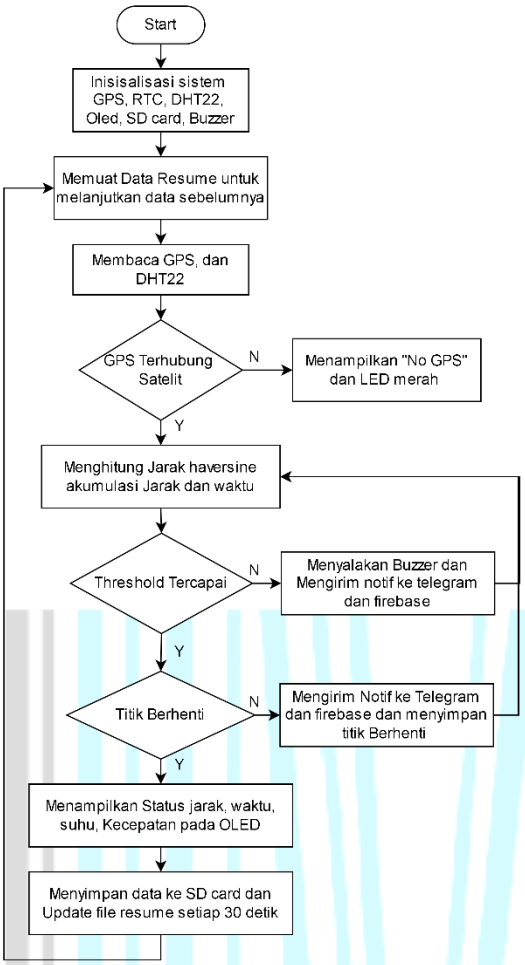
Gambar 2.2. Gambar Hardware

Pada Gambar 2.2 menampilkan purwarupa bentuk fisik perakitan perangkat keras yang telah dirangkai secara utuh dan ditempatkan ke dalam casing pelindung agar rapi, aman dari debu/cuaca, dan siap untuk dilakukan uji coba lapangan pada kendaraan.

2.2 Perancangan Software

Perangkat lunak dikembangkan menggunakan Arduino IDE berbahasa C/C++ dengan arsitektur dual-core FreeRTOS: Core 1 menjalankan loop utama (GPS, sensor, OLED, SD Card, buzzer) secara realtime tanpa gangguan, sedangkan Core 0 menjalankan WiFi task secara paralel di background (koneksi Telegram, pengiriman antrian notifikasi, polling perintah bot).

Perhitungan jarak menggunakan Formula Haversine: $d = 2R \times \arctan2(\sqrt{a}, \sqrt{(1-a)})$ dengan $a = \sin^2(\Delta lat/2) + \cos(lat_1)\cdot\cos(lat_2)\cdot\sin^2(\Delta lon/2)$, R = 6.371 km. Deteksi titik berhenti menggunakan metode berbasis radius 20 meter (STOP_RADIUS_M = 20 m) selama 30 detik (STOP_CONFIRM_S = 30 s) yang lebih andal dibandingkan metode berbasis kecepatan GPS yang rentan terhadap noise saat sinyal lemah. Data persisten disimpan ke file RESUME.dat sehingga tidak hilang saat alat dimatikan dan dilanjutkan secara otomatis saat dinyalakan kembali.



Gambar 2.3. Flowchart Sistem

Pada Gambar 2.3 menunjukkan diagram alir (*flowchart*) urutan logika pelaksanaan program pada mikrokontroler ESP32, mulai dari proses inisialisasi periferal, pembacaan data sensor secara berkala, kalkulasi jarak tempuh harian, hingga algoritma penentuan titik pemberhentian kurir.

Berdasarkan Gambar 2.3, alur kerja dari sistem monitoring kendaraan kurir ini dibagi menjadi beberapa tahapan logis sebagai berikut:

1. **Tahap Inisialisasi Sistem (*Start & Initialization*):**

Saat alat pertama kali dinyalakan (*power on*), mikrokontroler ESP32 akan melakukan inisialisasi terhadap seluruh komponen periferal yang terhubung. Proses ini meliputi pengaktifan komunikasi I2C untuk layar OLED dan RTC DS3231, inisialisasi jalur SPI untuk modul SD Card, pembacaan sensor DHT22, serta pembukaan koneksi serial untuk modul GPS Neo-6M. Jika seluruh komponen berhasil terinisialisasi, *buzzer* akan berbunyi sebagai indikator bahwa sistem siap beroperasi.

2. **Tahap Akusisi Data (*Data Acquisition*):**

Setelah inisialisasi berhasil, sistem masuk ke dalam siklus utama. Modul GPS secara terus-menerus menangkap sinyal satelit untuk mendapatkan koordinat bumi (*latitude* dan *longitude*) serta kecepatan kendaraan. Secara bersamaan, RTC DS3231 menyediakan data waktu (jam, menit, detik) dan tanggal yang presisi, sementara sensor DHT22 membaca suhu lingkungan di sekitar kotak penyimpanan paket.

3. **Tahap Kalkulasi Jarak Tempuh (*Odometer Tracking*):**

Sistem akan memeriksa validitas data koordinat dari GPS. Jika data koordinat valid dan terdeteksi adanya perubahan posisi dari titik sebelumnya, ESP32 akan menghitung selisih jarak menggunakan Formula Haversine. Hasil perhitungan jarak antar-titik tersebut kemudian diakumulasikan ke dalam variabel jarak total harian (berfungsi sebagai odometer digital). Informasi jarak tempuh, waktu, dan suhu ini akan diperbarui dan ditampilkan secara *real-time* pada layar OLED.

4. **Tahap Logika Deteksi Titik Berhenti (*Stop Point Detection*):**

Algoritma sistem secara konstan memeriksa kondisi pergerakan kendaraan kurir. Sistem menerapkan parameter *threshold* spasial dan temporal (radius < 20 meter dan durasi waktu $\ge$ 30 detik). Jika kendaraan berada di dalam radius sempit tersebut selama lebih dari 30 detik, sistem akan mengidentifikasi aktivitas tersebut sebagai **Titik Berhenti (Stop Point)** yang valid (baik untuk keperluan pengantaran paket, tertahan di lampu merah, atau kemacetan jalan).

5. **Tahap Penyimpanan Data dan Notifikasi (*Logging & Notification*):**

Ketika sebuah titik berhenti terdeteksi, mikrokontroler akan mengeksekusi dua perintah utama:

- **Penyimpanan Lokal:** Data koordinat presisi, waktu berhenti, suhu kotak, dan akumulasi jarak tempuh langsung ditulis ke dalam *file* log di dalam SD Card.
- **Penyimpanan Cloud/Notifikasi:** Data tersebut dikirimkan dalam bentuk pesan teks otomatis menuju bot Telegram admin perusahaan melalui jaringan internet sebagai bentuk pemantauan jarak jauh.

6. **Looping Kembali (*End of Cycle*):**

Setelah data berhasil disimpan dan dikirim, sistem akan mengosongkan timer berhenti dan kembali ke tahap akusisi data GPS untuk memantau pergerakan kendaraan pada segmen rute berikutnya. Proses ini berulang secara terus-menerus selama alat mendapatkan suplai daya.

3. HASIL DAN PEMBAHASAN

3.1 Pengujian Akurasi Modul GPS Neo-6M

Pengujian GPS dilakukan untuk memverifikasi kemampuan modul dalam menerima sinyal satelit dan menghasilkan data koordinat yang valid. Pengujian dilakukan di beberapa kondisi berbeda meliputi area terbuka, pinggir gedung, dan dalam ruangan.

Tabel 2. Hasil pengujian akurasi GPS Neo-6M

No	Lokasi	Satelit	TTFF (det)	Akurasi (m)	Status
1	Dibawah pohon	8	32	±3,2	Valid ✓
2	Di Jalan	9	28	±2,8	Valid ✓
3	Dalam ruangan 1 lantai	4	47	±4,1	Valid ✓
4	Dalam Gedung >2 lantai	-	-	-	No Signal
5	Dibawah Underpass	-	-	-	No Signal
6	Area terbuka	10	25	±2,5	Valid ✓

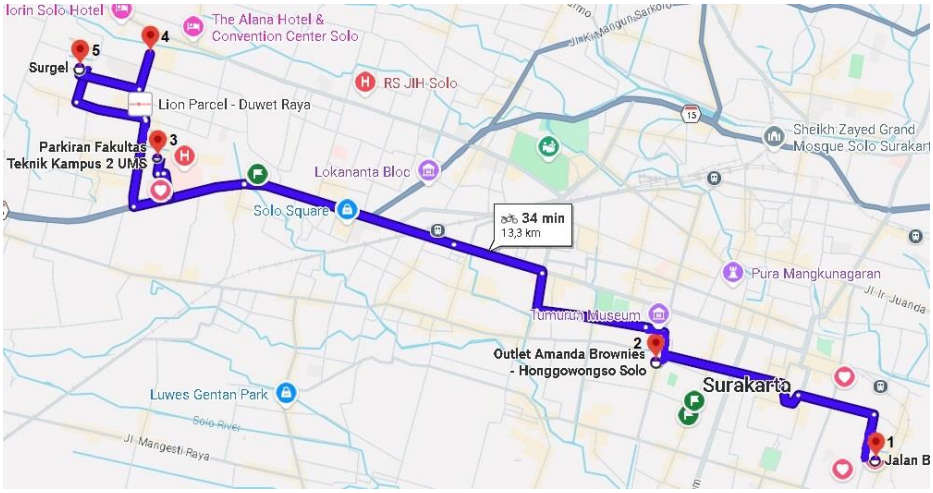
Hasil menunjukkan modul mampu mengunci sinyal rata-rata 8 satelit di area terbuka dengan TTFF rata-rata 33 detik pada kondisi cold start dan akurasi posisi 2,5–4,1 meter. Penurunan akurasi pada kondisi terhalang gedung merupakan karakteristik umum GPS sipil.

3.2 Pengujian Jarak Tempuh Point-to-Point

Pengujian dilakukan pada 6 rute berbeda di wilayah Kota Surakarta. Jarak referensi diukur menggunakan Google Maps. Seluruh pengujian mencatat pembacaan suhu DHT22 dan waktu RTC yang valid.

Tabel 3. Hasil pengujian jarak tempuh point-to-point

No	Titik Awal	Titik Akhir	Alat (km)	Real (km)	Selisih (km)	Error (%)	Suhu
1	-7.555161, 110.771247	-7.582320, 110.835761	7,80	8,10	0,30	3,70	<40°
2	-7.547350, 110.764357	-7.555161, 110.771247	1,72	1,67	0,05	2,99	<40°
3	-7.547350, 110.764357	-7.573582, 110.816313	7,38	7,50	0,12	1,60	<40°
4	-7.573582, 110.816313	-7.582320, 110.835761	3,22	2,86	0,36	12,59*	<40°
5	-7.582320, 110.835761	-7.547350, 110.764357	10,38	10,32	0,06	0,58	<40°
6	-7.547350, 110.764357	-7.545993, 110.770743	1,05	1,01	0,04	3,96	<40°
Total			31,55	31,46	0,93	25,42	
Rata -rata			**5,26**	**5,25**	**0,155**	**4,24**	



Gambar 3.1.Rute asli titik ke titik

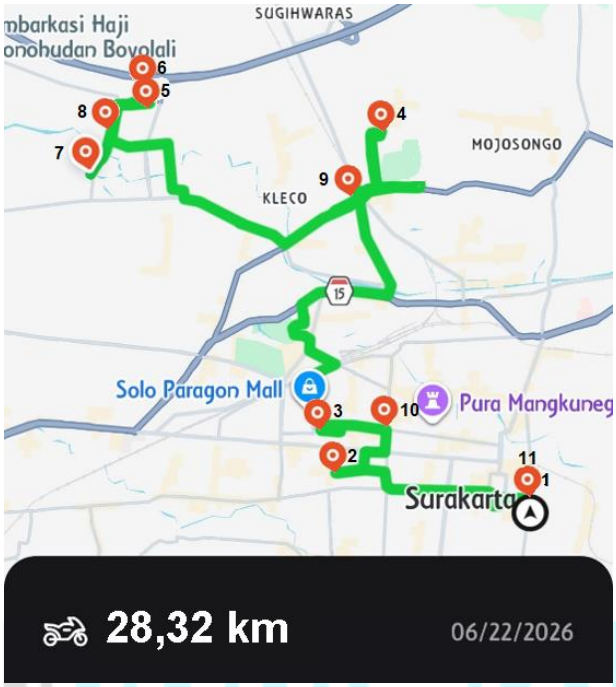
Rata-rata error keseluruhan sebesar 4,24%, dengan nilai terkecil 0,58% (Rute 5) dan terbesar 12,59% pada Rute 3 yang ditandai tanda bintang (*). Error tinggi pada Rute 3 diperkirakan disebabkan oleh efek *multipath* sinyal GPS akibat pepohonan dan bangunan di sepanjang rute permukiman tersebut, serta akumulasi error pada jalan berbelok yang tidak sepenuhnya terepresentasikan oleh interval sampling 1 detik. Jika Rute 3 dikecualikan sebagai pencilan, rata-rata error turun menjadi 2,57%, yang tergolong sangat baik untuk aplikasi monitoring armada skala operasional. Untuk rute per point nya dapat dilihat pada lampiran (Lampiran 1-6).

3.3 Pengujian Multi-Point Titik Berhenti

Pengujian multi-point dilakukan pada rute simulasi pengiriman paket di wilayah Surakarta dan sekitarnya dengan mencatat seluruh titik berhenti secara otomatis menggunakan metode radius 20 meter selama 30 detik. Pengujian berlangsung dalam dua sesi (siang dan sore) dengan total 11 titik berhenti yang berhasil terdeteksi dan dikirimkan ke Telegram secara otomatis.

Tabel 4. Hasil pengujian multi-point titik berhenti kurir

No	Lokasi Titik Berhenti	Jam	Jarak Alat (km)	Jarak Segmen (km)
1	-7.582319, 110.835737	10:25	0,02	—
2	-7.570009, 110.813996	10:41	3,33	3,31
3	-7.565407, 110.812740	10:51	5,22	1,89
4	-7.534113, 110.824282	11:16	10,08	4,86
5	-7.540758, 110.827047	11:50	16,87	6,79
6	-7.525110, 110.797890	11:07	17,38	0,51
7	-7.523683, 110.799562	16:39	18,84	1,46
8	-7.554799, 110.771526	16:44	20,33	1,49
9	-7.543735, 110.770354	16:55	22,45	2,12
10	-7.555985, 110.804188	17:11	26,62	4,17
11	-7.571688, 110.829824	17:17	28,25	1,63



Gambar 3.2. Rute Real Pengujian Multi Point

Pada titik berhenti ke-9 (Stadion Manahan), jarak kumulatif alat sebesar 22,45 km dibandingkan jarak referensi 22,42 km menghasilkan selisih hanya 0,03 km dengan error 0,13%. Nilai ini menunjukkan akurasi kumulatif yang sangat tinggi setelah menempuh 9 titik berhenti. Jeda panjang antara titik 6 (11:07) dan titik 7 (16:39) merupakan waktu istirahat/jeda sesi pengujian dan tidak memengaruhi keakuratan pencatatan karena data disimpan secara persisten di RESUME.dat.

Pada metode deteksi titik berhenti yang diatur menggunakan radius 20 meter selama batas waktu 30 detik, kendaraan yang tidak bergerak akibat lampu merah atau kondisi jalan yang macet total berpotensi terekam sebagai "titik berhenti" (stop point). Jika kendaraan tertahan di satu area (beradius < 20 meter) selama lebih dari durasi threshold 30 detik tersebut, sistem akan menganggapnya valid sebagai titik peristirahatan/pemberhentian. Dalam konteks operasional riil di jalanan perkotaan yang padat, fenomena ini tidak dapat dihindari sepenuhnya. Untuk mengatasi anomali tersebut, admin perusahaan kurir dapat membandingkan data koordinat berhenti ini dengan kondisi jalan secara logis.

Sistem Telegram Bot berhasil mengirimkan notifikasi otomatis untuk setiap titik berhenti yang terkonfirmasi, termasuk koordinat GPS dalam format link Google Maps, waktu, jarak tempuh, rata-rata kecepatan segmen. Seluruh 11 notifikasi titik berhenti terkirim dengan rata-rata waktu respons 1,8 detik sejak terkonfirmasi.

3.4. Pengujian Database Online (Firebase Realtime Database)

Pengujian *database* dilakukan untuk memverifikasi tingkat keberhasilan sinkronisasi data telemetri dari perangkat keras (ESP32) menuju server awan (*cloud server*) Firebase Realtime Database. Berbeda dengan metode pemrosesan sekuensial biasa, sistem IoT ini dikonfigurasi menggunakan arsitektur *Dual-Core* berbasis FreeRTOS. Tugas utama pembacaan sensor GPS, perhitungan rumus Haversine, interaksi layar OLED, dan penyimpanan SD Card diisolasi penuh pada **Core 1**. Sementara itu, fungsi konektivitas WiFi, bot Telegram, serta protokol pengiriman data Firebase (wifiTask) dialokasikan pada **Core 0**. Pembagian tugas ini memastikan pengiriman data yang lambat atau fluktuasi sinyal internet tidak memicu rugi-rugi data (*data loss*) pada pencatatan parameter kurir harian secara *real-time*.

Protokol pengiriman data menuju Firebase diimplementasikan melalui arsitektur REST API dengan memanfaatkan pustaka HTTPClient tanpa beban memori tambahan. Komunikasi data dilakukan menggunakan metode enkapsulasi JSON yang dikirimkan melalui operasi pengiriman HTTP PUT dan HTTP PATCH. Struktur penyimpanan data pada Firebase Realtime Database dirancang secara hierarkis per sesi harian dengan format penguncian kunci (*session key*) YYYYMMDD_NamaKurir.

Secara umum, terdapat lima fungsi utama pengiriman data ke server online yang diuji:

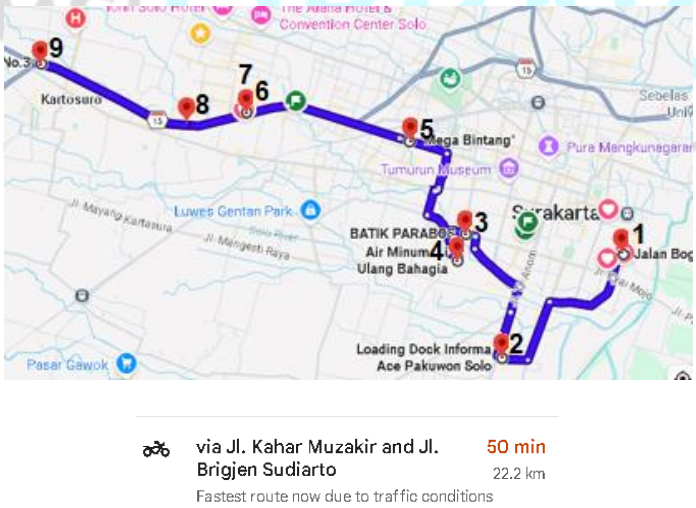
1. **Inisialisasi Sesi (firebaseInisialisasiSesi)**: Mengirimkan metadata awal berupa nama kurir, tanggal operasional, waktu mulai bekerja, dan status ke dalam *node* /sesi/{sessionKey}/info.
2. **Log Tracking Data (firebaseKirimLogTracking)**: Mengirimkan runtunan data telemetri berkala (seperti *timestamp*, koordinat garis lintang/bujur, total jarak, kecepatan, suhu, kelembaban, dan jumlah satelit) ke dalam *node* /sesi/{sessionKey}/tracking/{idx} setiap interval 5 detik (Lampiran 7).
3. **Pembaruan Ringkasan Sesi (firebaseUpdateRingkasan)**: Melakukan penulisan ulang (*overwrite*) terhadap akumulasi data jarak tempuh harian, durasi total, jumlah titik berhenti kurir, serta koordinat lokasi terakhir ke dalam *node* /sesi/{sessionKey}/ringkasan (Lampiran 8).
4. **Log Titik Berhenti (firebaseKirimTitikBerhenti)**: Merekam riwayat lokasi tempat kurir berhenti secara otomatis berbasis perhitungan radius ke dalam *node* /sesi/{sessionKey}/titik_berhenti/{no}, lengkap dengan estimasi sisa bahan bakar minyak (BBM) dan tautan otomatis menuju Google Maps (Lampiran 9).

5. **Detak Jantung Alat (firebaseHeartbeat)**: Mengirimkan sinyal penanda status keaktifan alat (*heartbeat status*) secara berkala ke dalam *node* /alat/{Nama_Kurir}/status untuk memantau indikator kekuatan sinyal RSSI WiFi dan keaktifan alat dari jarak jauh.

Hasil pencatatan pengujian fungsionalitas pengiriman muatan data (*data payload*) dari serial monitor perangkat menuju kluster data Firebase Realtime Database dirangkum pada Tabel 5 berikut.

Tabel 5. Hasil pengujian Database multi-point titik berhenti kurir

No	Jam	Koordinat (Latitude, Longitude)	Jarak Alat (km)	Kec. Rata-rata Segmen	Selisih Jarak (km)	Persentase Kesalahan (%)
1	13:12	-7.582367, 110.835702	0,011	0,0 km/jam	—	—
2	13:23	-7.599703, 110.815255	3,829	26.3 km/jam	—	—
3	13:31	-7.578959, 110.809113	7,550	33,2 km/jam	—	—
4	13:34	-7.583477, 110.807788	8,655	24,4 km/jam	—	—
5	13:50	-7.563511, 110.799610	14,191	22,5 km/jam	—	—
6	13:55	-7.555734, 110.772115	17,755	40,9 km/jam	—	—
7	13:56	-7.556663, 110.772201	17,784	6,4 km/jam	—	—
8	13:57	-7.558644, 110.772152	18,059	11,7 km/jam	—	—
9	14:03	-7.550497, 110.737513	22,224	56,2 km/jam	—	—
10	14:30	-7.560755, 110.761960	25,319	42,0 km/jam	—	—
11	14:32	-7.557288, 110.769313	26,370	30,0 km/jam	—	—
12	14:38	-7.556632, 110.780439	28,129	18,6 km/jam	**+0,329**	**1,18%**
-	**Total**	**Jarak Akhir (Dashboard)**	**28,170**	—	**+0,370**	**1,33%**



Gambar 3.3. Rute Real Pengujian Multipoint Database (point 1-9)

Gambar 3.4. Rute Real Pengujian Multipoint Database (point 10-12)

Berdasarkan hasil pengujian rute dinamis (multi-point) nyata sepanjang 27,8 km, akumulasi pembacaan odometer pada alat mencatat angka 28,170 km dengan persentase kesalahan (error rate) sebesar 1,33%. Performa waktu proses perhitungan jarak menggunakan algoritma Haversine pada ESP32 terbukti sangat akurat dalam membaca pergeseran koordinat pelacakan, di mana waktu komputasinya lebih unggul dan presisi jika dihadapkan pada kontur pergerakan bumi dibandingkan menggunakan perhitungan Euclidean konvensional (Miftahuddin, Umaroh and Karim, 2020).

Simpangan data jarak sebesar 370 meter tersebut adalah hal yang sangat wajar (memenuhi batas toleransi) akibat efek GPS drift atau kesalahan fluktuasi satelit navigasi saat memperbarui sistem koordinat ketika kendaraan berada di bawah objek rintangan atau area padat bangunan (Setiadi *et al.*, 2023).

3.4 Pengujian Integrasi Telegram Bot IoT

Sistem Telegram Bot diuji dengan mengirimkan seluruh perintah yang tersedia (/start, /status, /lokasi, /suhu, /laporan, /titikberhenti, /clearsavepoint, /reset) dan memantau respons sistem melalui hotspot ponsel. Seluruh perintah berhasil direspons dengan rata-rata waktu respons 1,8 detik dan akurasi 100%. Notifikasi threshold jarak, waktu, dan suhu terkirim otomatis saat kondisi terpenuhi. Mekanisme antrian pesan (queue) pada Core 0 memastikan tidak ada notifikasi yang hilang saat koneksi terputus sementara.

3.5 Rekap Data Pengujian Keseluruhan

Tabel berikut merangkum seluruh hasil pengujian sistem monitoring kendaraan kurir yang telah dilakukan:

Tabel 6. Rekap hasil pengujian keseluruhan sistem

Parameter Pengujian	Hasil Sistem	Referensi	Error (%)	Status
Rata-rata error jarak P2P (6 rute)	4,24%	≤ 10%	—	✓
Error terkecil (Rute 5, Rumah–Kost)	0,58%	—	—	✓
Error terbesar (Rute 3, kondisi GPS terganggu)	12,59%	—	—	**Catatan***
Rata-rata error P2P tanpa pencilan	2,57%	≤ 5%	—	✓

Akurasi kumulatif multi-point (9 titik)	22,45 km	22,42 km	0,13%	✓
Titik berhenti multi-point terdeteksi	11 / 11	11 / 11	100%	✓
Pencatatan suhu DHT22 (valid)	6 / 6	100%	0%	✓
Pencatatan waktu RTC (valid)	6 / 6	100%	0%	✓
Penyimpangan waktu RTC (24 jam)	≤ +3 detik	≤ ±10 detik	—	✓
Keberhasilan notifikasi buzzer	100%	100%	0%	✓
Delay respons buzzer	±119 ms	≤ 500 ms	—	✓
Respons perintah Telegram Bot	100%	100%	—	✓
Rata-rata waktu respons Telegram	1,8 detik	≤ 5 detik	—	✓
Konsumsi daya rata-rata sistem	±280 mA	≤ 500 mA	—	✓
Estimasi runtime baterai 2×18650	±22 jam	≥ 8 jam	—	✓

*) Pengujian Rute 3 point to point menghasilkan error 12,59% yang dipengaruhi gangguan sinyal GPS di area permukiman padat. Pengujian yang lain menunjukkan konsistensi yang baik dengan rata-rata error 2,57% jika data pencilan dikecualikan.

Pada Tabel 6 ini merupakan rekapitulasi komprehensif dari seluruh evaluasi yang dilakukan terhadap kinerja sistem, mencakup tingkat akurasi lokasi, validitas sensor, latensi pengiriman pesan Telegram, dan efisiensi konsumsi daya dibandingkan dengan standar referensi yang diharapkan.

4. PENUTUP

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, dapat disimpulkan bahwa:

1. Pengujian jarak point-to-point pada 6 rute menghasilkan rata-rata error 4,24%. Pengujian multi-point menunjukkan akurasi kumulatif 0,13% setelah menempuh 9 titik berhenti.
2. Metode deteksi titik berhenti berbasis radius 20 meter selama 30 detik terbukti lebih andal dibandingkan metode berbasis kecepatan GPS yang rentan terhadap noise. Seluruh 11 titik berhenti pada pengujian multi-point berhasil terdeteksi dan dikirimkan ke Telegram secara otomatis.
3. Sistem persisten data (RESUME.dat) memastikan akumulasi jarak dan waktu tidak hilang saat alat dimatikan, dan hanya direset melalui tombol fisik (3x tekan) atau perintah /reset Telegram.
4. Alat ini Mambantu Perusahaan kurir dalam hal transparansi rute dan efisiensi logistik. Alat ini dapat menekan potensi penyalahgunaan armada untuk keperluan pribadi di luar rute kerja, mencegah manipulasi jarak tempuh untuk klaim bensin, serta mempermudah evaluasi kinerja para kurir karena data disajikan dengan akurat dan tersimpan secara berkesinambungan.

5. Alat ini Membantu Kurir mendapatkan perlindungan administratif karena jarak yang ditempuh direkam otomatis sebagai bukti riil dan otentik kinerjanya. Hal ini akan mencegah kerugian jika klaim jarak pengiriman diragukan perusahaan. Selain itu, adanya *buzzer* sebagai notifikasi otomatis mempermudah kurir untuk mengetahui peringatan atau batas threshold tertentu tanpa harus terpaku menatap layar instrumen secara manual selama berkendara.

Untuk pengembangan selanjutnya, disarankan penambahan filter Kalman pada data GPS untuk meningkatkan akurasi pada kondisi sinyal tidak stabil, dan pengujian pada armada kendaraan lebih besar untuk menilai skalabilitas sistem.

DAFTAR PUSTAKA

Adiyanto, F.C., Rintyarna, B.S. and Ariyani, S. (2026) 'Sistem Deteksi Kecelakaan Motor Dengan Notifikasi Telegram Berbasis Sensor MPU6050 Dan GPS', 8, pp. 164–175.

Fariza, M., Hasanuddin, T. and Umar, F. (2022) 'Implementasi Metode Haversine Formula untuk Menentukan Jarak Terdekat pada Pengantaran Air Galon Depot Anantama Berbasis Android', 3(1), pp. 69–78.

Mau'idzoh Hasanah, Carudin, I.P. (2023) 'Implementasi Algoritma Haversine Pada Fitur Geolocation Aplikasi Pencarian Terapis Baby Spa Mau'idzoh', 9(15), pp. 376–390.

Miftahuddin, Y., Umaroh, S. and Karim, F.R. (2020) 'PERBANDINGAN METODE PERHITUNGAN JARAK EUCLIDEAN , HAVERSINE , (STUDI KASUS : INSTITUT TEKNOLOGI NASIONAL BANDUNG)', 14(2), pp. 69–77.

Pardede, R., Farisi, H. and Arwani, I. (2017) 'Pengembangan Sistem Aplikasi Monitoring Sepeda Motor Berbasis IoT dengan Modul GPS Guna Pemantauan dan Keamanan Kendaraan (Studi kasus : Roganda Rental Motorbike)', 1(1), pp. 1–10.

Prasetyo, L.A., Himawan, I. and Sihombing, R.A. (2026) 'SISTEM DETEKSI DINI DAN PELACAKAN PENCURIAN KENDARAAN BERMOTOR MENGGUNAKAN GPS NEO-6M DAN PLATFORM ANDROID', 3(4), pp. 550–558.

Rohman, S. and Amalia, Z. (2025) 'Rancang Bangun Sistem Monitoring Dan Keselamatan Dengan Fitur Live Tracking Pada Mobil Rental'.

S.Samsugi, F.A.A. (2024) 'Sistem Keamanan Kendaraan Bermotor Dengan ESP32 Menggunakan Kontrol Android', 14(1), pp. 169–181.

Setiadi, B. *et al.* (2023) 'Estimasi Jarak pada Sistem Koordinat Berbasis Metode Haversine menggunakan Tapis Kalman', 11(1), pp. 207–216.

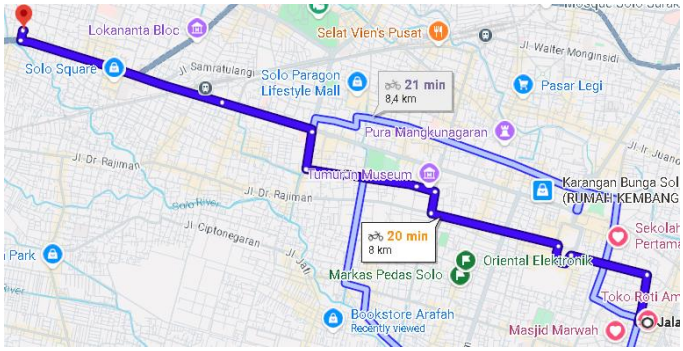
Suppa, R. and Wahyuni, V.I. (2024) 'RANCANG BANGUN ALAT PELACAK POSISI KENDARAAN BERBASIS IoT', 3(1), pp. 29–33.

Tafrikhatin, A., Wulandari, A.T. and Fauzi, M. (2021) 'Rancang Sistem GPS Tracker Pada

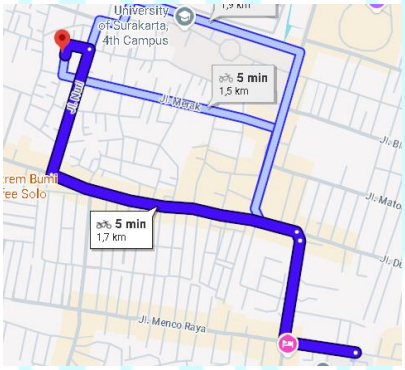
Motor Berbasis Arduino', 1(2), pp. 114–119.

Tsaqib, M. *et al.* (2025) 'Rancang Bangun Sistem Keamanan Kendaraan 2WD Berbasis ESP 32 dengan Autentikasi Sidik Jari dan Pelacakan Lokasi melalui Telegram', pp. 1034–1042.

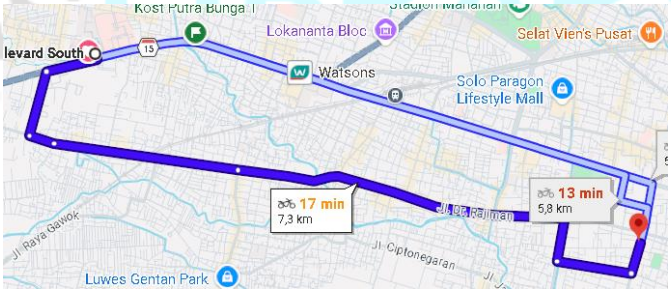
LAMPIRAN



Gambar 1. Rute Point 1

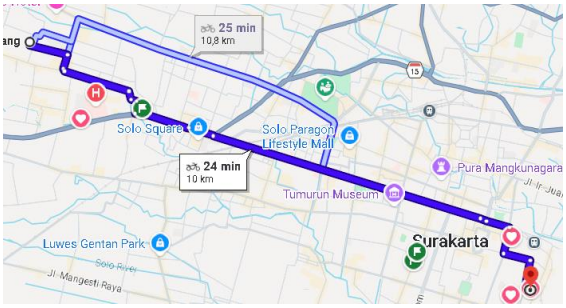


Gambar 2. Rute Point 2

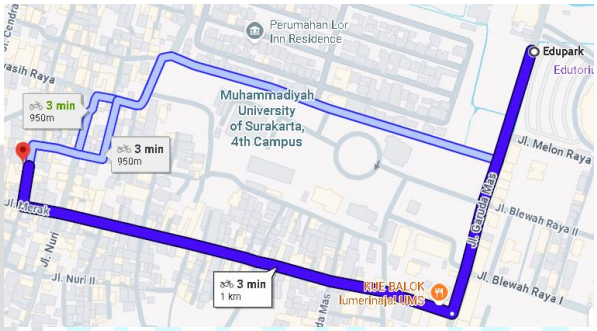


Gambar 3. Rute Point 3

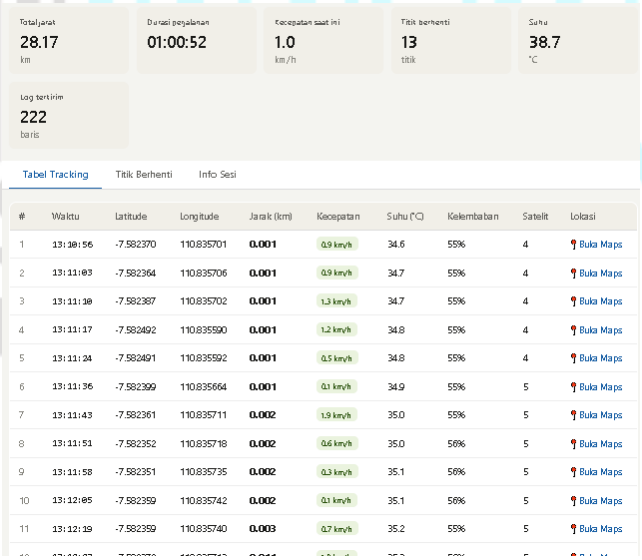
Gambar 4. Rute Point 4



Gambar 5. Rute Point 5



Gambar 6. Rute Point 6



Total jarak	Durasi perjalanan	Kecepatan saat ini	Titik berhenti	Suhu
28.17 km	01:00:52	1.0 km/h	13 titik	38.7 °C

Log terkirim: 222 baris

Tabel Tracking | Titik Berhenti | Info Sesi

#	Waktu	Latitude	Longitude	Jarak (km)	Kecepatan	Suhu (°C)	Kelembaban	Satelit	Lokasi
1	13:10:56	-7.582370	110.835701	0.001	0.9 km/h	34.6	55%	4	Buka Maps
2	13:11:03	-7.582364	110.835706	0.001	0.9 km/h	34.7	55%	4	Buka Maps
3	13:11:10	-7.582387	110.835702	0.001	1.3 km/h	34.7	55%	4	Buka Maps
4	13:11:17	-7.582492	110.835590	0.001	1.2 km/h	34.8	55%	4	Buka Maps
5	13:11:24	-7.582491	110.835592	0.001	0.5 km/h	34.8	55%	4	Buka Maps
6	13:11:36	-7.582399	110.835664	0.001	0.1 km/h	34.9	55%	5	Buka Maps
7	13:11:43	-7.582361	110.835711	0.002	1.9 km/h	35.0	55%	5	Buka Maps
8	13:11:51	-7.582352	110.835718	0.002	0.6 km/h	35.0	56%	5	Buka Maps
9	13:11:58	-7.582351	110.835735	0.002	0.3 km/h	35.1	56%	5	Buka Maps
10	13:12:05	-7.582359	110.835742	0.002	0.1 km/h	35.1	56%	5	Buka Maps
11	13:12:19	-7.582359	110.835740	0.003	0.7 km/h	35.2	55%	5	Buka Maps

Gambar 7. Log Tracking Data

Tabel Tracking | Titik Berhenti | Info Sesi

Field	Nilai
Total jarak (km)	28.166
Durasi perjalanan	01:00:52
Kecepatan terakhir (km/h)	1
Jumlah titik berhenti	13
Suhu (°C)	38.7
Kelembaban (%)	45
Posisi terakhir — Lat	-7.55658
Posisi terakhir — Lon	110.780386
Update terakhir	14:19:40
Total baris log tracking	222

Gambar 8. Info Sesi

Tabel Tracking | Titik Berhenti | Info Sesi

#	Jam	Jarak saat berhenti (km)	Kec. rata-rata segmen	BBM sisa (L)	Latitude	Longitude	Lokasi
1	13:12	0.011 km	0.0 km/h	2.50 L	-7.582367	110.835702	Buka Maps
2	13:23	3.829 km	26.3 km/h	2.40 L	-7.599703	110.815255	Buka Maps
3	13:31	7.550 km	33.2 km/h	2.30 L	-7.578959	110.809113	Buka Maps
4	13:34	8.655 km	24.4 km/h	2.27 L	-7.583477	110.807788	Buka Maps
5	13:50	14.191 km	22.5 km/h	2.13 L	-7.563511	110.799610	Buka Maps
6	13:55	17.755 km	40.9 km/h	2.03 L	-7.556734	110.772115	Buka Maps
7	13:56	17.784 km	6.4 km/h	2.03 L	-7.556663	110.772201	Buka Maps
8	13:57	18.059 km	11.7 km/h	2.02 L	-7.558644	110.772152	Buka Maps
9	14:03	22.224 km	56.2 km/h	1.92 L	-7.550497	110.737513	Buka Maps
10	14:10	25.319 km	42.0 km/h	1.83 L	-7.560755	110.761960	Buka Maps
11	14:12	26.370 km	30.0 km/h	1.81 L	-7.557288	110.769313	Buka Maps
13	14:18	28.129 km	18.6 km/h	1.76 L	-7.556632	110.780439	Buka Maps

Gambar 9. Log Titik Berhenti