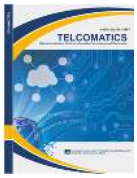
Available online at journal.uib.ac.id/index.php/telcomatics

TELCOMATICS

ISSN (online) 2541-5867



Perancangan dan Implementasi Sistem Keamanan Kendaraan Berbasis IoT Menggunakan ESP32, FastAPI, dan Aplikasi Android

Noe Prihartoyo Simanjuntak[1], M Abdur Rahman[1], Adi Nuzul Pratama[1], Stelyven[1], Andik Yulianto[2*]

[1]*Prodi Teknologi Informasi, Fakultas Ilmu Komputer, Universitas Internasional Batam, Jalan Gajah Mada, Baloi Sei Ladi, Batam, Indonesia*

[2]*Program Studi Sistem Informasi, Fakultas Ilmu Komputer, Universitas Internasional Batam*

ARTICLE INFO

Keywords:
Vehicle Security, IoT, ESP32, MQTT, FastAPI, Android, Firebase

Received: June 10, 2025
Revised: July 20, 2025
Accepted: July 25, 2025

***Corresponding author:**
E-mail: andik@uib.ac.id (Andik Yulianto)

DOI: 10.37253/telcomatics.v10i1.10870

ABSTRACT

The advancement of digital technology has made a major contribution to improving security systems, especially in the automotive sector. One of the main challenges in securing motor vehicles is providing real-time protection against potential threats such as theft or vandalism. In this study, an Internet of Things (IoT)-based vehicle security system was designed and implemented that utilizes the ESP32 microcontroller as the main tool to detect physical contact or interference with the vehicle. This system integrates the MQTT communication protocol for sending data to a FastAPI-based backend server, which then stores the data in MongoDB Atlas and sends notifications to users via Firebase Cloud Messaging (FCM). Users can monitor and respond to these notifications through an Android application built using Android Studio and Java programming language. In addition to notifications, this application also provides a sound alarm player feature and a display of sensor activity history. The test results show that the system is able to detect and respond to events quickly and accurately, with a notification delivery time of less than two seconds after the data is received by the server. With an integrated and real-time system architecture, this solution is expected to be an economical, efficient, and reliable alternative to improve the security of private and public vehicles.

I. PENDAHULUAN

Dalam era transformasi digital yang berkembang pesat, tantangan terhadap keamanan kendaraan bermotor turut mengalami pergeseran yang signifikan. Tidak hanya dari sisi fisik seperti penguncian konvensional, tetapi juga dari aspek pemantauan dan respon terhadap aktivitas mencurigakan yang terjadi di sekitar kendaraan. Di tengah meningkatnya angka pencurian kendaraan, baik roda dua maupun roda empat, kebutuhan akan sistem keamanan yang mampu memberikan perlindungan secara real-time menjadi semakin mendesak. Berbagai pendekatan telah dilakukan untuk menghadirkan solusi keamanan yang adaptif dan responsif, namun tidak semuanya mampu memberikan integrasi menyeluruh antara perangkat keras dan perangkat lunak yang efisien, hemat biaya, dan mudah digunakan [1].

Internet of Things (IoT) telah membuka peluang besar dalam pengembangan sistem keamanan kendaraan berbasis jaringan. IoT memungkinkan berbagai perangkat sensorik berkomunikasi melalui internet, mengumpulkan data dari lingkungan, dan mengirimkannya ke sistem pusat untuk diproses secara langsung. Dalam konteks keamanan kendaraan, teknologi ini memungkinkan sistem untuk mendeteksi keberadaan atau aktivitas fisik yang tidak diinginkan dan memberikan peringatan kepada pengguna dalam waktu singkat [2]. Mikrokontroler ESP32 menjadi salah satu komponen penting dalam pengembangan sistem ini karena kemampuannya untuk terhubung langsung ke jaringan Wi-Fi dan mendukung komunikasi dengan protokol ringan seperti MQTT (Message Queuing Telemetry Transport), yang sangat sesuai untuk kebutuhan sistem IoT berskala kecil dan menengah[3].

Selain perangkat keras, bagian penting lain dari sistem keamanan ini adalah bagaimana data yang diperoleh dari sensor dapat diproses, disimpan, dan diteruskan secara cepat. Untuk menjawab kebutuhan ini, digunakan arsitektur backend berbasis FastAPI, yaitu framework modern berbasis Python yang terkenal karena kecepatan dan efisiensinya dalam menangani permintaan RESTful API. *Server* ini tidak hanya bertugas menyimpan data ke basis data cloud MongoDB Atlas, tetapi juga mengelola logika bisnis yang menentukan apakah suatu aktivitas perlu dilaporkan kepada pengguna melalui notifikasi. Mekanisme pengiriman peringatan dilakukan melalui Firebase Cloud Messaging (FCM), yang memungkinkan *push notification* dikirim secara instan ke aplikasi Android yang telah didaftarkan[4].

Penelitian ini bertujuan untuk mengembangkan sistem keamanan kendaraan berbasis IoT yang terintegrasi secara penuh mulai dari sensor fisik hingga notifikasi ke aplikasi pengguna. Sistem yang dibangun tidak hanya mengandalkan satu jalur komunikasi seperti SMS atau Bluetooth, melainkan memanfaatkan infrastruktur internet dan *cloud computing* untuk memastikan skalabilitas dan fleksibilitas dalam implementasi di dunia nyata. Dengan membangun arsitektur sistem yang lengkap dan terbuka untuk pengembangan lebih

lanjut, diharapkan solusi ini dapat menjadi alternatif keamanan kendaraan yang efektif, terjangkau, dan mudah diadopsi oleh masyarakat luas.

II. KAJIAN PUSTAKA

Perkembangan teknologi Internet of Things (IoT) telah mendorong transformasi dalam berbagai sektor, termasuk dalam sistem keamanan kendaraan. Banyak studi telah menyoroti bagaimana integrasi antara perangkat sensor, jaringan internet, dan platform aplikasi dapat menciptakan sistem yang cerdas, efisien, dan mudah dioperasikan oleh pengguna. Pada bagian ini, akan diuraikan berbagai penelitian dan referensi yang relevan, yang mendasari rancangan sistem keamanan kendaraan berbasis IoT dengan ESP32, FastAPI, MQTT, MongoDB, dan aplikasi Android.

Pada penelitian [5] sistem keamanan kendaraan dikembangkan dengan menggunakan modul GPS dan GSM untuk mengirimkan lokasi kendaraan kepada pemilik saat terdeteksi adanya pencurian. Sistem ini bekerja dengan pendekatan pengiriman pesan teks melalui jaringan seluler. Meskipun cukup efektif, penggunaan SMS sebagai jalur komunikasi dinilai memiliki keterbatasan dari sisi biaya operasional dan kecepatan transmisi. Oleh karena itu, pendekatan menggunakan internet melalui IoT menjadi pilihan yang lebih efisien dan ekonomis, di mana protokol MQTT digunakan sebagai jalur komunikasi utama antara sensor dan *server* pusat. Protokol ini bersifat ringan dan efisien dalam konsumsi bandwidth, menjadikannya sangat cocok untuk perangkat seperti ESP32 yang memiliki keterbatasan sumber daya [2].

Lebih lanjut, pemanfaatan mikrokontroler ESP32 telah banyak diteliti sebagai solusi hemat biaya dan konsumsi daya rendah untuk sistem monitoring real-time. ESP32 mampu menangani pengiriman data sensor secara berkala tanpa mengalami latensi signifikan, terutama ketika dikombinasikan dengan protokol MQTT [6]. Hal ini memperkuat pilihan ESP32 sebagai pemroses utama perangkat keras dalam sistem yang sedang dikembangkan.

Pada sisi *server* penggunaan FastAPI sebagai *backend* dinilai lebih modern dan cepat dibandingkan framework tradisional seperti Flask atau Django. FastAPI dibangun menggunakan asynchronous programming dan tipe anotasi Python 3.7+, memungkinkan validasi input yang ketat serta performa yang kompetitif bahkan setara dengan Node.js dalam pengujian tertentu [4]. FastAPI juga mudah diintegrasikan dengan berbagai Pustaka Python lain seperti Pydantic dan motor (driver async MongoDB), yang memudahkan pengembangan sistem berskala kecil hingga menengah. Untuk penyimpanan data, MongoDB Atlas menjadi pilihan utama karena fleksibilitasnya dalam menangani data tidak terstruktur, serta kemudahan skalabilitasnya di *cloud*. MongoDB menunjukkan performa yang baik dalam menangani data sensor dari perangkat IoT, terutama dalam skenario real-time streaming data [7]. Struktur dokumennya memungkinkan data disimpan dengan format yang fleksibel, sehingga mendukung berbagai skenario pemrosesan yang dibutuhkan oleh sistem keamanan kendaraan.

Penggunaan Firebase Cloud Messaging (FCM) pada studi [8] menunjukkan efektivitas FCM dalam mengirimkan notifikasi push dengan latensi yang sangat rendah ke aplikasi Android. Notifikasi dapat dikustomisasi berdasarkan payload yang diterima *server*, memungkinkan sistem untuk memberikan peringatan secara langsung begitu terjadi interaksi mencurigakan pada sensor kendaraan.

Dari sisi aplikasi *mobile*, penelitian [9] menekankan pentingnya desain antarmuka yang sederhana namun informatif dalam aplikasi keamanan berbasis IoT. Aplikasi yang terlalu kompleks justru akan menyulitkan pengguna dalam mengambil keputusan cepat ketika terjadi ancaman. Dalam pengembangan sistem ini, prinsip antarmuka yang responsif dan ringan menjadi fokus utama. Integrasi keseluruhan antara perangkat keras, cloud *server*, dan antarmuka pengguna juga telah diteliti dalam [10], yang membangun sistem monitoring lingkungan rumah berbasis ESP32 dan aplikasi Android. Hasil dari studi tersebut menunjukkan bahwa ketika sistem dirancang dengan komunikasi dua arah yang baik serta alur data yang efisien, tingkat kepercayaan dan kepuasan pengguna meningkat secara signifikan. Beberapa literatur lain juga menegaskan pentingnya pemisahan antara komunikasi sensor dan manajemen pengguna. Misalnya, pada penelitian [10] agar komunikasi sensor menggunakan MQTT sedangkan autentikasi dan pengelolaan data pengguna menggunakan protokol HTTP/REST untuk keamanan dan efisiensi. Pendekatan ini diimplementasikan penuh dalam sistem yang dikembangkan dengan memisahkan jalur komunikasi sensor (MQTT) dan komunikasi pengguna (HTTP/Retrofit).

Dengan mengacu pada berbagai penelitian dan dokumentasi yang telah ada, sistem keamanan kendaraan yang dirancang tidak hanya dibangun atas dasar kebutuhan fungsional, tetapi juga mengacu pada praktik yang telah terbukti efektif di berbagai proyek IoT serupa. Hal ini memperkuat validitas rancangan dan membuka ruang untuk pengembangan lebih lanjut di masa depan.

Selain itu, penerapan teknologi kecerdasan buatan (AI) pada sistem keamanan kendaraan berbasis IoT juga semakin banyak diteliti dalam beberapa tahun terakhir. Algoritma *machine learning* yang diimplementasikan pada *server backend* mampu mendeteksi pola anomali pada data sensor. Pendekatan ini dapat meningkatkan keakuratan dalam mengidentifikasi upaya pencurian atau akses ilegal terhadap kendaraan [11]. Dengan memanfaatkan ESP32, FastAPI, dan MongoDB, sistem ini dapat mengolah data secara *real-time* serta menjalankan model prediksi dengan latensi yang rendah. Hal ini memperlihatkan potensi arsitektur IoT modern untuk mendukung analisis data yang lebih kompleks dalam konteks keamanan kendaraan.

Selain itu, aspek keamanan data dan privasi pengguna menjadi perhatian utama dalam pengembangan sistem IoT. pentingnya penerapan enkripsi *end-to-end* dan autentikasi multi-faktor pada aplikasi *mobile* untuk melindungi data sensitif [12]. Penggunaan protokol TLS/SSL untuk komunikasi HTTP serta penerapan token berbasis OAuth2 pada *backend* FastAPI. Strategi ini dinilai relevan untuk sistem yang sedang dikembangkan, karena dapat memastikan komunikasi antara perangkat ESP32, server, dan aplikasi Android tetap aman dari serangan man-in-the-middle maupun penyadapan.

Sementara itu, pada penelitian [13] memperkenalkan konsep edge computing dalam sistem keamanan kendaraan

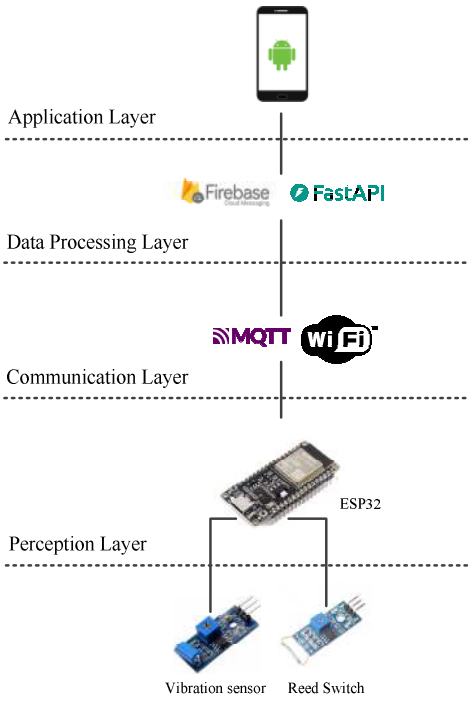
berbasis IoT sebagai solusi untuk mengurangi ketergantungan pada server cloud serta mempercepat waktu respon. Dengan memanfaatkan kemampuan pemrosesan lokal pada ESP32, sebagian proses analisis dapat dilakukan langsung pada perangkat sebelum data dikirim ke *server* pusat melalui MQTT. Pendekatan ini terbukti efektif, terutama pada kondisi jaringan yang kurang stabil, sehingga sistem tetap mampu mengirimkan peringatan secara cepat melalui FCM dan aplikasi Android.

III. Metode Penelitian

Dalam penelitian ini sistem keamanan kendaraan dirancang menggunakan pendekatan arsitektur berlapis Internet of Things (IoT) yang terdiri dari *Perception Layer, Network and Communication Layer, Data Processing Layer* dan *Application Layer*. Setiap lapisan memiliki peran krusial dalam memastikan sistem dapat mendeteksi gangguan secara fisik pada kendaraan dan menyampaikannya secara cepat ke pengguna melalui jaringan internet. Sistem ini bertujuan untuk memberikan notifikasi instan terhadap getaran atau pergerakan yang tidak diinginkan pada kendaraan, sekaligus memberikan kontrol kepada pengguna untuk mengatur mode keamanan melalui aplikasi Android.

A. Diagram Sistem

Adapun diagram sistem yang dirancang untuk perangkat IoT pengaman kendaraan penelitian ini ditunjukkan pada Gambar 1. Berikut adalah penjelasan dari tiap lapisan (*layer*).



Gambar 1. Diagram Blok Sistem

1. Perception Layer
   Lapisan ini merupakan komponen awal dari sistem IoT yang berfungsi sebagai pengumpul data dari lingkungan fisik, dalam hal ini dari kendaraan yang dilindungi. Dua jenis sensor utama digunakan untuk mendeteksi potensi gangguan:
   - Sensor SW-420 (Vibration Sensor): Sensor ini digunakan untuk mendeteksi getaran fisik yang terjadi pada kendaraan. Sensor bekerja berdasarkan prinsip perubahan tegangan akibat adanya getaran, dan akan mengeluarkan sinyal digital HIGH apabila getaran melebihi ambang batas tertentu. Sensor ini sensitif terhadap guncangan kecil seperti seseorang menyentuh atau memukul kendaraan.
   - Reed Switch (Magnetic Sensor): Sensor magnetik yang akan aktif (ON atau tertutup) saat berada dekat dengan medan magnet, dan non-aktif (OFF atau terbuka) saat magnet dijauhkan. Reed Switch berfungsi sebagai sensor keamanan yang mendeteksi apakah jok kendaraan dibuka atau tidak.

   Kedua sensor ini dihubungkan dengan mikrokontroler ESP32, yang memproses sinyal digital dari sensor dan meneruskannya ke lapisan berikutnya. ESP32 diprogram untuk memantau status sensor secara terus-menerus dan mengirimkan data ke server apabila ada aktivitas yang terdeteksi.

2. Network and Communication Layer
   Lapisan ini menjadi jembatan antara data fisik dari perangkat keras dan pemrosesan sistem cloud, yang memungkinkan transfer data secara efisien dan aman. Beberapa komponen dan protokol utama pada lapisan ini meliputi:
   - ESP32 dengan Modul Wi-Fi: Digunakan untuk menghubungkan perangkat ke jaringan internet dan mengirimkan data secara langsung menggunakan protokol MQTT.
   - Protokol MQTT (Message Queuing Telemetry Transport): Protokol komunikasi yang digunakan untuk mengirimkan data sensor ke broker publik broker.emqx.io. MQTT menggunakan mekanisme publish-subscribe, di mana ESP32 bertindak sebagai publisher dan server backend FastAPI sebagai subscriber.

   Seluruh komponen dalam lapisan ini bekerja secara sinkron untuk memastikan bahwa aktivitas dari sensor segera diterjemahkan menjadi notifikasi kepada pengguna dalam waktu yang sangat singkat.

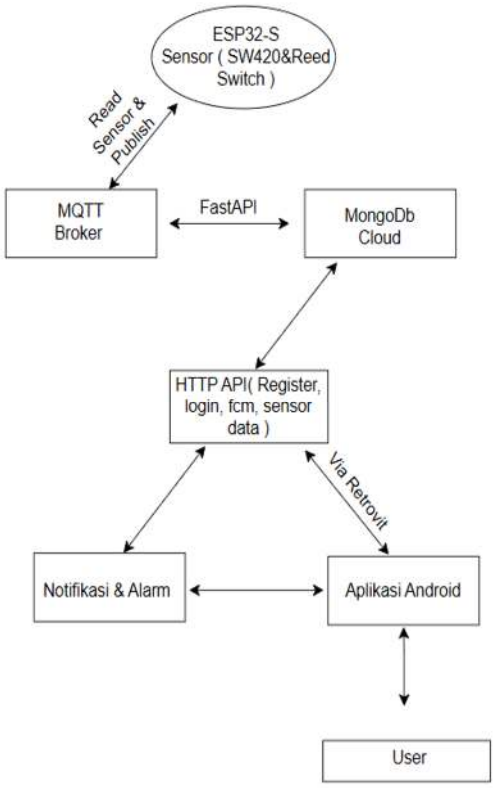
3. Data Processing Layer
   Lapisan ini berfungsi untuk mengolah data yang dikirim oleh sensor. Server Backend FastAPI: Framework Python modern yang bertanggung jawab untuk menangani data sensor yang diterima dari MQTT, menyimpannya ke MongoDB Atlas, serta menjalankan logika keamanan seperti pengiriman notifikasi.
   MongoDB Atlas: Basis data NoSQL berbasis cloud yang digunakan untuk menyimpan data sensor, data pengguna, dan log aktivitas. Struktur JSON memungkinkan penyimpanan data yang fleksibel dan scalable.
   Firebase Cloud Messaging (FCM): Digunakan untuk mengirimkan push notification secara real-time ke

perangkat pengguna. Server backend mengirim pesan ke Firebase saat data sensor menunjukkan adanya gangguan. Retrofit: Digunakan untuk melakukan komunikasi HTTP antara aplikasi dan backend, khususnya untuk login, registrasi, dan pengambilan data.

4. Application Layer
   Lapisan ini menjadi penghubung antara sistem dan pengguna akhir, memungkinkan pengguna untuk memonitor status kendaraan dan merespons gangguan dengan cepat melalui aplikasi pada smartphone. Dibangun dengan Android Studio menggunakan bahasa Java. Aplikasi terdiri dari beberapa fitur utama, yakni login, registrasi, pengaturan mode keamanan (Security Mode dan alarm mode), settingan alarm, dan riwayat aktivitas sensor.

B. Flowchart Sistem

Gambar 2 menjelaskan alur kerja sistem, hubungan antar layer secara menyeluruh, mulai dari inisialisasi dan deteksi sensor hingga proses monitoring dan kontrol melalui aplikasi android. Adapun penjelasan mengenai setiap tahap diagram alir sebagai berikut.



Gambar 2. Diagram aliran data sistem keamanan kendaraan

Alur dimulai dari tahap inisialisasi perangkat, di mana mikrokontroler ESP32 melakukan proses booting dan koneksi ke jaringan Wi-Fi. Setelah koneksi internet berhasil, ESP32 selanjutnya menginisialisasi sensor yang terhubung, yaitu sensor getar SW-420 dan reed switch (sensor magnetik). SW-420 bertugas mendeteksi getaran fisik pada kendaraan seperti pukulan atau guncangan, sementara reed switch mendeteksi perubahan posisi magnet, misalnya ketika jok kendaraan dibuka secara paksa. Kedua sensor ini bekerja secara paralel dan dipantau secara terus-menerus dalam sebuah loop pemeriksaan status sensor.

Jika tidak terdapat aktivitas mencurigakan, ESP32 akan terus menjalankan proses monitoring secara diam-diam tanpa mengirim data. Namun, ketika salah satu dari kedua sensor mendeteksi adanya gangguan—baik berupa getaran maupun pembukaan jok—maka ESP32 segera mengirimkan data peringatan dalam bentuk payload JSON melalui protokol MQTT ke broker MQTT publik (broker.emqx.io). Data ini berisi informasi penting seperti status sensor, timestamp, dan identitas perangkat. Selanjutnya, server backend yang dibangun menggunakan FastAPI berperan sebagai subscriber dari topik MQTT tertentu dan secara otomatis menerima data tersebut. Begitu data diterima, *server* akan memprosesnya dan menyimpannya ke dalam basis data MongoDB Atlas dengan struktur yang telah ditentukan. Setelah data berhasil tersimpan, *server* akan melakukan pengecekan nilai sensor yang diterima. Apabila data tersebut mengindikasikan potensi ancaman atau gangguan nyata, maka *server* akan mengaktifkan modul Firebase Cloud Messaging (FCM) untuk mengirimkan notifikasi instan ke perangkat Android milik pengguna.

Dari sisi pengguna, aplikasi Android yang telah dipasang dan terhubung dengan akun pengguna melalui proses login akan menerima notifikasi tersebut dalam waktu singkat. Pengguna akan mendapatkan pemberitahuan berbentuk pop-up yang dapat langsung diakses. Saat notifikasi diterima, sistem juga memicu pemutaran suara alarm secara otomatis melalui aplikasi, dengan menggunakan file suara yang telah dipilih sebelumnya oleh pengguna dari 10 opsi alarm yang tersedia. Pengguna juga dapat secara manual mematikan alarm atau mengganti suara alarm melalui pengaturan aplikasi. Selain menerima notifikasi, pengguna juga memiliki akses penuh untuk melakukan kontrol terhadap sistem, seperti mengaktifkan atau menonaktifkan mode keamanan kendaraan melalui fitur "Save Mode" dan "Security Mode". Semua interaksi pengguna ini dikirim kembali ke *server* melalui protokol HTTP menggunakan Retrofit di sisi Android dan diproses oleh FastAPI di backend. Sebagai pelengkap, pengguna juga dapat melihat riwayat aktivitas sensor yang ditampilkan dalam format waktu yang telah diolah secara manusiawi, memberikan konteks yang lebih informatif terhadap log keamanan kendaraan.

Secara keseluruhan, diagram alir sistem ini tidak hanya menggambarkan aliran logika program dan data dari sensor hingga aplikasi, tetapi juga mencerminkan bagaimana sistem ini dirancang dengan prinsip responsif, modular, dan real-time. Proses yang terjadi di setiap node diagram alir mewakili sinergi antara perangkat keras, perangkat lunak, dan infrastruktur jaringan yang saling melengkapi untuk mencapai sistem keamanan kendaraan yang cerdas, terjangkau, dan mudah dioperasikan.

C. Keamanan Sistem

Keamanan sistem dalam proyek Vehicle Security System berbasis IoT ini dirancang tidak hanya untuk mendeteksi ancaman fisik terhadap kendaraan, tetapi juga untuk memastikan bahwa data, komunikasi, dan otorisasi pengguna

terjaga dengan baik. Implementasi keamanan dilakukan di berbagai lapisan sistem, mulai dari keamanan autentikasi pengguna, enkripsi data, pemisahan jalur komunikasi sensor dan pengguna, hingga pengamanan komunikasi *cloud-to-device.*

Dari sisi keamanan identitas pengguna, sistem menerapkan mekanisme autentikasi berbasis email dan password. Pada saat proses registrasi melalui aplikasi Android, pengguna diminta untuk memasukkan informasi email, password, dan nomor plat kendaraan. Data ini kemudian dikirim ke *server* backend melalui jalur HTTP menggunakan Retrofit. Sebelum disimpan ke database MongoDB Atlas, password pengguna tidak disimpan dalam bentuk plaintext, melainkan melalui proses pengacakan (hashing) menggunakan algoritma bcrypt. Penggunaan bcrypt menjadi salah satu standar keamanan yang disarankan dalam [14] karena sifatnya yang tahan terhadap serangan *brute force* dan mampu mengatur tingkat kompleksitas hash melalui *cost factor*. Dengan demikian, jika sewaktu-waktu database diretas, *password* pengguna tidak dapat langsung diambil atau digunakan secara langsung oleh pihak tidak bertanggung jawab. Dari sisi komunikasi data, sistem menggunakan dua jalur komunikasi yang terpisah, komunikasi antara sensor dan *server* dilakukan melalui protokol MQTT, sedangkan komunikasi antara aplikasi Android dan *backend* menggunakan protokol HTTP melalui Retrofit.

Pemisahan ini penting untuk mencegah *overlapping* trafik serta menjaga efisiensi dan keamanan sistem, sebagaimana disarankan oleh [15]. MQTT dipilih untuk komunikasi sensor karena protokol ini lebih ringan, hemat bandwidth, dan memiliki overhead kecil dibandingkan protokol lain seperti HTTP. Sementara untuk komunikasi pengguna, protokol HTTP dirancang agar mudah diatur ke mode HTTPS saat sistem dideploy, yang berarti data yang dikirim akan dienkripsi secara otomatis oleh sertifikat SSL/TLS, sehingga meminimalisir risiko *sniffing* atau penyadapan selama transmisi data di jaringan terbuka.

Keamanan sistem juga dijaga melalui mekanisme pengiriman notifikasi push yang bersifat personal dan terkontrol, menggunakan Firebase Cloud Messaging (FCM). Setiap perangkat Android yang telah login akan menerima token unik dari Firebase, dan hanya token tersebut yang akan menerima notifikasi jika terjadi deteksi aktivitas mencurigakan dari sensor. Mekanisme ini tidak hanya mengurangi kemungkinan pengiriman notifikasi ke perangkat yang tidak berwenang, tetapi juga mencegah terjadinya broadcast tidak sah yang dapat mengganggu kenyamanan pengguna. Sistem ini sejalan dengan pendekatan yang dilakukan pada [16] dalam mengelola komunikasi satu arah dari *cloud* ke *device* secara aman dan efisien. Pendekatan keamanan juga diperkuat melalui struktur dan manajemen database yang baik. MongoDB Atlas sebagai sistem penyimpanan data mendukung fitur-fitur seperti kontrol akses berbasis user, whitelist IP, serta koneksi terenkripsi menggunakan TLS. Penelitian [17] menunjukkan bahwa MongoDB merupakan solusi penyimpanan yang handal dalam proyek IoT berskala kecil hingga menengah karena fleksibilitas skema dokumennya serta fitur keamanan *cloud* yang lengkap. Dalam proyek ini, database dikonfigurasi untuk hanya menerima koneksi dari *server* backend, dan bukan langsung dari aplikasi, guna mencegah akses data secara langsung dari sisi klien.

Dari aspek arsitektur sistem, proyek ini dirancang secara modular dan terpisah sesuai dengan arsitektur 3-layer IoT. Desain berlapis memperkuat segmentasi keamanan dan mencegah satu titik kerentanan berdampak pada keseluruhan sistem. Arsitektur berlapis merupakan salah satu pendekatan paling efektif dalam membangun sistem IoT yang aman dan *scalable* [17]. Semua komponen sistem diuji secara integratif untuk memastikan bahwa skenario-skenario keamanan dapat ditangani dengan benar. Saat sensor mendeteksi ancaman, hanya pengguna yang terautentikasi yang dapat menerima notifikasi, membuka riwayat sensor, atau mengakses pengaturan alarm. Tidak ada data sensitif pengguna yang disimpan secara lokal di aplikasi, dan sesi pengguna dikelola secara aman melalui singleton UserSession.

Dengan berbagai pendekatan ini, sistem keamanan kendaraan yang dibangun tidak hanya fokus pada proteksi fisik kendaraan, tetapi juga memastikan bahwa setiap aspek teknis—baik dari sisi pengguna, perangkat keras, komunikasi, hingga penyimpanan *cloud*—telah dilindungi dengan prinsip dan teknologi keamanan yang andal serta sesuai dengan praktik terbaik yang telah diterapkan dalam banyak sistem IoT modern.

D. Perancangan Sistem

Pada bagian ini, dijelaskan secara rinci mengenai beberapa aspek

1. Perancangan Perangkat Keras (*Perception Layer*)
   Pada bagian ini, dijelaskan secara rinci mengenai perangkat keras yang digunakan dalam sistem.
2. Perancangan Jalur Komunikasi dan *Server* (*Network &Communication Layer*)
   Setelah data fisik diperoleh, langkah berikutnya adalah merancang bagaimana data tersebut dikirim dan diolah. Pada sistem ini, arsitektur komunikasi menggunakan dua jalur utama yaitu MQTT untuk komunikasi sensor ke *server* dan HTTP (via Retrofit) untuk komunikasi aplikasi ke *server*. ESP32 dikonfigurasi sebagai publisher MQTT yang akan mengirim data ke broker broker.emqx.io, sebuah broker MQTT publik yang efisien dan andal. *Server* backend dibangun menggunakan FastAPI, yang bertindak sebagai subscriber dan sekaligus pengelola data. Saat *server* menerima data dari sensor, ia akan menyimpannya ke dalam MongoDB Atlas, yang merupakan database NoSQL cloud-based. Sesuai saran dari Fernandes & Diniz (2022), pemisahan jalur komunikasi ini memperkuat keamanan dan efisiensi karena komunikasi sensor dan pengguna berjalan secara independen.
3. Perancangan Antarmuka Aplikasi Android (Application Layer)
   Bagian perancangan ini mencakup tampilan dan fungsionalitas aplikasi Android sebagai media control utama dari pengguna. Semua interaksi aplikasi dikirim melalui HTTP menggunakan Retrofit dan ditangani oleh API FastAPI. Aplikasi juga dilengkapi Firebase Cloud Messaging untuk menerima notifikasi otomatis saat sensor mendeteksi ancaman.
4. Perancangan Basis Data

Bagian ini menjelaskan struktur data dan penyimpanannya dalam MongoDB Atlas. Basis data dibagi menjadi dua koleksi utama yaitu Koleksi users untuk menyimpan data pengguna seperti email, password (yang telah di-hash dengan bcrypt), dan plat nomor dan Koleksi sensor untuk menyimpan log aktivitas dari sensor (timestamp, status sensor, ID perangkat,).

5. Rancangan Keamanan Sistem
   Pada bagian ini dijelaskan bahwa sistem menerapkan Bcrypt untuk hashing password pengguna, Token FCM unik per perangkat, Whitelist IP untuk database MongoDB, Desain modular yang memisahkan fungsi sensor, *server*, dan aplikasi. Dengan pendekatan ini, sistem tidak hanya mendeteksi ancaman fisik, tetapi juga menjaga integritas dan keamanan data secara digital.

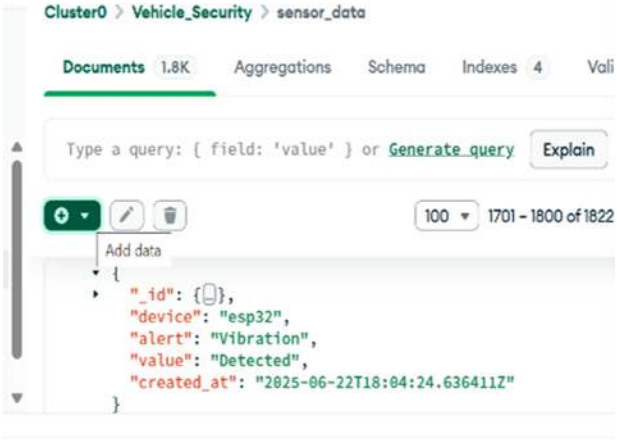
E. Implementasi Sistem

Implementasi dari sistem keamanan kendaraan berbasis Internet of Things (IoT) ini dilakukan secara bertahap, dimulai dari pembangunan infrastruktur perangkat keras hingga pengembangan perangkat lunak dan komunikasi antar-komponen yang berfungsi secara sinkron. Sistem yang telah dirancang sebelumnya kemudian diterapkan pada prototipe nyata untuk menguji keterpaduan antara sensor fisik, mikrokontroler, jaringan komunikasi, serta aplikasi pemantauan dan kendali berbasis Android.

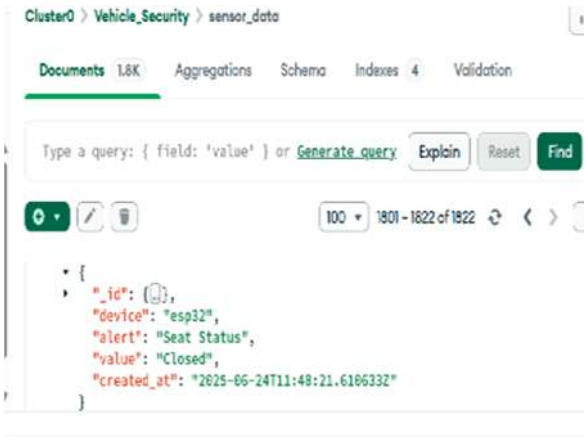
Proses implementasi diawali dengan penyusunan rangkaian perangkat keras yang terdiri dari sensor SW-420 sebagai pendeteksi getaran dan reed switch sebagai pendeteksi pembukaan jok kendaraan. Kedua sensor ini dipasang secara strategis pada bodi dan bagian jok kendaraan guna memastikan bahwa setiap gangguan fisik yang mencurigakan dapat terdeteksi secara akurat. Sensor-sensor tersebut dihubungkan ke mikrokontroler ESP32 yang memiliki kapabilitas koneksi Wi-Fi, dan diprogram melalui Arduino IDE untuk membaca kondisi sensor secara terus menerus. Dalam pemrograman tersebut, ESP32 dikonfigurasikan agar dapat mengirimkan data ke broker MQTT (broker.emqx.io) setiap kali mendeteksi adanya perubahan status sensor. Data yang dikirim berupa format JSON yang berisi informasi waktu, status sensor, dan identitas perangkat, serta disampaikan melalui topik khusus yang telah didefinisikan. Rangkaian perangkat keras yang diimplementasikan pada kendaraan dapat dilihat pada Gambar 3.

Gambar 3. Rangkaian prototipe dan implementasi sensor pada jok sepeda motor

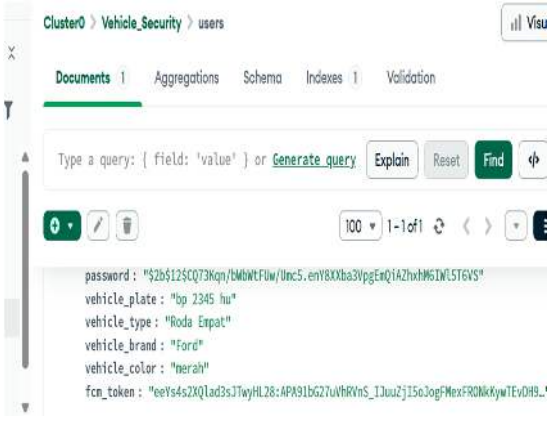
Pada Gambar 4 menunjukkan bahwa sensor yang mendeteksi akan tersimpan kedalam database *cloud* MongoDB Atlas menggunakan driver async motor. Data dari sensor yang diterima oleh *server* akan dianalisis untuk menentukan apakah kondisi tersebut merupakan potensi gangguan atau hanya aktivitas normal.



Gambar 4. Deteksi sensor Vibration



Gambar 5. Deteksi Reed Switch (Magnetic Sensor)

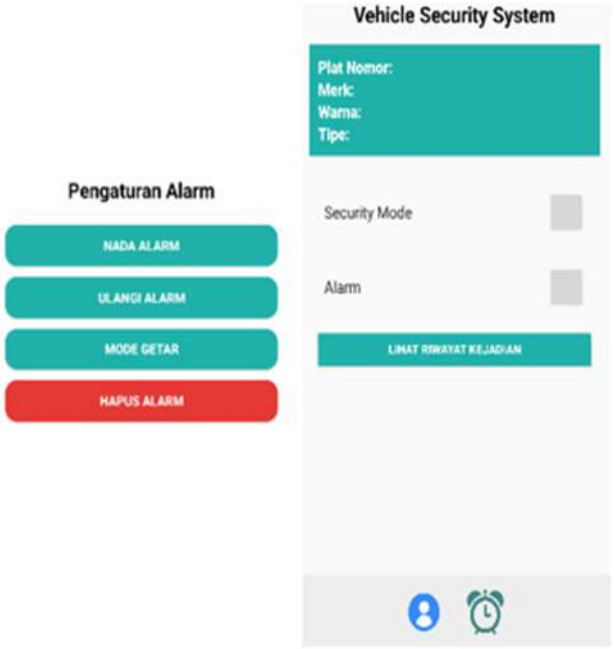


Gambar 6. Data User hasil dari registrasi pada aplikasi dan disimpan pada database di koleksi Use

 https://doi.org/10.37253/telcomatics.v10i1.10870

Untuk pengguna juga sudah disiapkan koleksi user untuk menyimpan informasi dari user yang telah melakukan registrasi dan di integrasikan kecloud database MongoDB. Data akan ditampilkan seperti pada gambar 5

Pada sisi pengembangan aplikasi android, digunakan android studio dengan Bahasa pemograman java. Terdapat beberapa fitur penting seperti settingan alarm yang didalamnta terdapat 10 media player untuk dipakai oleh user. Fitur penting lain seperti security mode dan Riwayat deteksi sensor yang digunakan utuk mengontrol dan memonitoring kendaraan. Gambar 5 menunjukkan bagaimana tampilan UI (User Interface) pada pengembangan aplikasi android.



Gambar 7. Fitur Utama Security Mode, Riwayat Sensor dan (b) Pengaturan Alarm

IV. HASIL DAN PEMBAHASAN

Hasil implementasi sistem keamanan kendaraan berbasis IoT menunjukkan bahwa seluruh komponen dapat bekerja secara terpadu dan responsif. Sensor SW-420 dan reed switch yang terpasang pada kendaraan berhasil mendeteksi gangguan fisik secara akurat, seperti getaran pada stang yang di goyangkan oleh orang asing dalam melakukan tindak penucrian pada umumnya atau pembukaan jok, dan mengirimkan data ke *server* melalui protokol MQTT dengan waktu pengiriman rata-rata kurang dari dua detik.

Mikrokontroler ESP32 mampu membaca perubahan status sensor dan mempublikasikan data secara real-time ke broker MQTT tanpa jeda signifikan. Aplikasi Android yang dikembangkan mampu menerima notifikasi dengan stabil, memutar suara alarm secara otomatis, serta memberikan kontrol kepada pengguna untuk mengaktifkan atau menonaktifkan Alarm dan Security Mode. Selain itu, fitur riwayat aktivitas sensor berjalan dengan baik, menampilkan data log yang terstruktur berdasarkan waktu kejadian dalam format yang mudah dipahami. Gambar 8 menunjukkan bagaimana tampilan data sensor yang tersimpan di Aplikasi android. Untuk notifikasi yang muncul juga ditampilkan di bar atas aplikasi/hp yang terkirim secara real time pada saat sensor mendeteksi. Tampilan notifikasi bisa dilihat dan ditampilkan pada Gambar 9.

Gambar 8. Tampilan Riwayat Data Sensor Yang mendeteksi

Gambar 9. Tampilan Notifikasi Realtime pada Aplikasi atau bar atas perangkat user

V. KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, sistem keamanan kendaraan berbasis Internet of Things (IoT) yang dibangun dalam proyek ini terbukti mampu menjalankan fungsi utamanya secara optimal, yaitu mendeteksi ancaman terhadap kendaraan secara fisik dan memberikan respons cepat melalui notifikasi serta alarm digital. Sistem mengintegrasikan beberapa komponen penting, seperti sensor SW-420 untuk deteksi getaran dan reed switch sebagai sensor pembukaan jok, yang dikendalikan oleh mikrokontroler ESP32 dengan konektivitas Wi-Fi. Mekanisme komunikasi data.

Dengan menggunakan protokol MQTT memungkinkan pengiriman informasi sensor secara ringan, cepat, dan efisien ke *server* backend berbasis FastAPI. *Server* berhasil mengelola data secara real-time, menyimpan informasi ke dalam database MongoDB Atlas yang fleksibel, serta mengaktifkan notifikasi instan menggunakan Firebase Cloud Messaging (FCM) kepada pengguna. Aplikasi Android yang dikembangkan tidak hanya mampu menerima notifikasi secara stabil, tetapi juga menyediakan antarmuka intuitif untuk kontrol sistem dan pemantauan riwayat aktivitas sensor, serta memungkinkan pengguna memilih dan memutar alarm suara sebagai bentuk respons terhadap potensi ancaman.

Dari aspek keamanan, sistem telah menerapkan prinsip perlindungan data melalui *hashing password* menggunakan algoritma bcrypt dan pemisahan jalur komunikasi untuk sensor dan pengguna. Hal ini sejalan dengan prinsip keamanan data pada sistem IoT yang menekankan kerahasiaan, integritas, dan ketersediaan. Meskipun demikian, proyek ini masih memiliki tantangan, terutama dalam hal ketergantungan terhadap koneksi

Jaringan dan potensi false alarm akibat sensitivitas sensor yang tinggi, yang memerlukan kalibrasi lebih lanjut. Secara keseluruhan, sistem yang dirancang dan diimplementasikan dalam proyek ini dapat dijadikan sebagai model awal dari solusi keamanan kendaraan berbasis IoT yang terjangkau, responsif, dan mudah diakses oleh pengguna. Dengan pengembangan lanjutan, seperti penambahan sistem otentikasi dua faktor, penyempurnaan algoritma deteksi, dan perluasan konektivitas melalui cloud publik, sistem ini berpotensi menjadi platform keamanan kendaraan yang cerdas dan adaptif untuk kebutuhan di era digital yang semakin terhubung.

Dengan menerapkan pendekatan arsitektur tiga lapis meliputi lapisan persepsi, komunikasi, dan aplikasi sistem ini menunjukkan keunggulan desain modular yang dapat memperkuat daya tahan dan kemudahan pengembangan dalam penggunaan nyata. Pembagian fungsi di setiap lapisan mempermudah proses perawatan maupun peningkatan fitur di kemudian hari, seperti penambahan pelacakan lokasi menggunakan GPS, deteksi gerakan berbasis kamera, atau integrasi dengan sistem kota pintar. Fleksibilitas ini memungkinkan sistem untuk digunakan pada berbagai tipe kendaraan, mulai dari kendaraan pribadi, kendaraan instansi pemerintah, hingga armada transportasi umum.

VI. DAFTAR PUSTAKA

[1] A. Putra and D. Romahadi, "Sistem Keamanan Sepeda Motor Berbasis Internet Of Things (Iot) Dengan Smartphone Menggunakan Nodemcu," 2021.

[2] M. Juliarto, R. Amru Nityasa, and A. D. Fajar Aditama, "Perancangan Keamanan Kendaraan Tanpa Kunci Dengan Menggunakan ESP32 dan Aplikasi BLYNK Berbasis IOT," *V-MAC (Virtual of Mechanical Engineering Article)*, vol. 9, no. 1, pp. 47–53, Mar. 2024, doi: 10.36526/v-mac.v9i1.3653.

[3] F. D. Makatita and N. F. A. Hakim, "MQTT Protocol-Based ESP-32 Smarthome with Multi-sensor Recognition," *Journal of Electrical, Electronic, Information, and Communication Technology*, vol. 6, no. 1, p. 29, May 2024, doi: 10.20961/jeeict.6.1.84007.

[4] I. Liwanto, H. Arfandy, A. Munir, and S. 63, "Jurnal Ilmu Komputer KHARISMA TECH PENDISTRIBUSIAN INFORMASI DI STMIK KHARISMA MAKASSAR."

[5] A. Mustafa *et al.*, "Vehicle Intrusion And Theft Control System Using GSM and GPS -- An advance and viable approach," Apr. 2020, doi: 10.1109/ICACC.2009.154.

[6] M. Adrian *et al.*, "REAL-TIME DATA ACQUISITION WITH ESP32 FOR IOT APPLICATIONS USING OPEN-SOURCE MQTT BROKERS," *Proceedings in Manufacturing Systems*, vol. 19, pp. 61–68, 2024, [Online]. Available: https://www.researchgate.net/publication/388464048

[7] P. Mahardika Kusumawardhana, M. Hannats, H. Ichsan, and R. Primananda, "Implementasi Penyimpanan Data Sensor Nirkabel dengan MongoDB pada Lingkungan IOT Menggunakan Protokol MQTT," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 9, pp. 3391–3399, Feb. 2018, Accessed: Aug. 23, 2025. [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/2291

[8] B. Wicaksono, A. Susanto, T. Informatika, F. Ilmu Komputer, and U. Dian Nuswantoro Semarang, "Push Notification Using Firebase Cloud Messaging (FCM) on Employee Attendance Application," *SISFOTENIKA*, vol. 11, no. 2, pp. 220–231, Aug. 2021, doi: 10.30700/JST.V11I2.1150.

[9] E. Sandalci, "A CASE STUDY ON DEFINING SIMPLICITY IN INTERACTION DESIGN: A REVIEW OF THE SCHEMATIC AND FUNCTIONAL EVALUATION OF THE SIMPLE MODE SMARTPHONE INTERFACE," 2021.

[10] M. A. Khan *et al.*, "Smart Android Based Home Automation System Using Internet of Things (IoT)," *Sustainability 2022, Vol. 14, Page 10717*, vol. 14, no. 17, p. 10717, Aug. 2022, doi: 10.3390/SU141710717.

[11] R. B. Varugu, G. Anil Kumar, and R. Supervisor, "A Survey on IoT Device Authentication and Anomaly Detection for Cyber Security using Machine Learning." [Online]. Available: https://ssrn.com/abstract=4798899

[12] S. K. V, T. C. Manjunath, and A. Professor, "Design & Implementation of data privacy & security using IoT sensors in remote health monitoring system [1]," 2023.

[13] Y. H. Chang, F. C. Wu, and H. W. Lin, "Design and Implementation of ESP32-Based Edge Computing for Object Detection," *Sensors (Basel)*, vol. 25, no. 6, p. 1656, Mar. 2025, doi: 10.3390/S25061656.

[14] T. P. Batubara, S. Efendi, and E. B. Nababan, "Analysis Performance BCRYPT Algorithm to Improve Password Security from Brute Force," *J Phys Conf Ser*, vol. 1811, no. 1, p. 012129, Mar. 2021, doi: 10.1088/1742-6596/1811/1/012129.

[15] S. Hong, J. Kang, and S. Kwon, "Performance Comparison of HTTP, HTTPS, and MQTT for IoT Applications," *International journal of advanced smart convergence*, vol. 12, no. 1, pp. 9–17, 2023, doi: 10.7236/IJASC.2023.12.1.9.

[16] B. Wicaksono, A. Susanto, T. Informatika, F. Ilmu Komputer, and U. Dian Nuswantoro Semarang, "Push Notification Using Firebase Cloud Messaging (FCM) on Employee Attendance Application," *SISFOTENIKA*, vol. 11, no. 2, pp. 220–231, Aug. 2021, doi: 10.30700/JST.V11I2.1150.

[17] P. Mahardika Kusumawardhana, M. Hannats, H. Ichsan, and R. Primananda, "Implementasi Penyimpanan Data Sensor Nirkabel dengan MongoDB pada Lingkungan IOT Menggunakan Protokol MQTT," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 9, pp. 3391–3399, Feb. 2018, Accessed: Aug. 24, 2025. [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/2291