BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Berdasarkan kajian terhadap penelitian terdahulu yang relevan dengan topik monitoring kebisingan dan getaran berbasis *IoT*, diperoleh berbagai pendekatan teknologi serta temuan penting yang mendukung pengembangan sistem pada penelitian ini. Ringkasan penelitian – penelitian tersebut disajikan secara sistematis dalam Tabel 2.1 untuk memudahkan perbandingan focus penelitian, teknologi yang digunakan, serta kontribusi utama masing-masing studi

Tabel 2. 1 Penelitian Terkait

Peneliti dan Tahun	Judul Penelitian	Teknologi dan Sensor	Hasil Penelitian	Keterbatasan Penelitian Terdahulu
A. Singh et al. (2024) [9]	*Real-time vibration monitoring and analysis of agricultural tractor drivers*	Sistem berbasis IoT, sensor getaran (WBV), transmisi	Nilai rata-rata akselerasi WBV A(8) sebesar 0.64 m/s² untuk	Fokus pada sektor pertanian dan parameter getaran saja; belum mengintegrasikan dashboard web

	using an IoT-based system	data ke cloud	periode kerja 8 jam	interaktif serta sistem notifikasi otomatis
Bhima Sankar Manthina et al. (2025) [10]	*IoT-based Noise Monitoring using Mobile Nodes for Smart Cities*	ESP32, DFRobot analog sound sensor (SEN0232), node bergerak	Tingkat kebisingan rata-rata >75 dB(A), dengan beberapa kejadian >90 dB(A)	Fokus pada pemetaan kebisingan perkotaan; belum membahas integrasi dashboard web komprehensif dan evaluasi risiko spesifik
Lia Wilani et al. (2023) [32]	Rancang Bangun Sistem Monitoring Kebisingan Pada Ruangan Dengan Sensor GY-MAX4466 Berbasis IoT	Sensor suara GY-MAX4466, sistem IoT, aplikasi antarmuka pengguna	Akurasi 97,58% dan presisi 97,84% dalam pemantauan kebisingan real-time	Terbatas pada satu parameter (kebisingan); belum mendukung monitoring multi-parameter dan analisis

				risiko berbasis web terintegrasi
Aditya Rinaldi & M. Anang Jatmiko (2025) [33]	Monitoring Real Time Tingkat Kebisingan Laboratorium *Ship Power Plan* Berbasis IoT	Sensor GY-MAX4466, ESP WROOM-32, website monitoring	Akurasi >90% dan latensi rendah dalam monitoring kebisingan real-time	Hanya memantau kebisingan; belum mengintegrasikan getaran serta mekanisme notifikasi dini berbasis web
I. Turkin et al. (2023) [34]	*The Use of IoT for Determination of Time and Frequency Vibration Characteristics of Industrial Equipment for Condition-Based Maintenance*	MEMS accelerometer, sistem IoT nirkabel	Nilai maksimum kecepatan getaran (Max RMS) sebesar 0.35 m/s	Terbatas pada analisis karakteristik getaran; belum mengintegrasikan monitoring multi-parameter dan visualisasi dashboard web terpusat

2.2 Landasan Teori

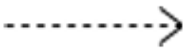
2.2.1 *Unified Modelling Languange* (UML)

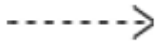
Dalam Pengembangan perangkat lunak, dibutuhkan sebuah bahasa pemodelan standar yang dapat digunakan untuk menggambarkan, merancang, dan mendokumentasikan sistem secara visual, yang dikenal dengan istilah *Unified Modeling Language* (UML). Dengan menggunakan notasi grafis yang terstandarisasi, UML memudahkan komunikasi antara pengembang, analisis sistem, dan pemangku kepentingan dalam memahami struktur maupun alur kerja sistem yang sedang dibangun. Dalam penelitian ini terdapat empat jenis diagram UML yang digunakan, yaitu :

1. *Use Case* Diagram

   Dalam tahap analisis kebutuhan sistem, dibutuhkan sebuah diagram yang mampu menggambarkan interaksi antara pengguna dengan sistem secara jelas dan mudah dipahami. Diagram ini merepresentasikan kebutuhan fungsional sistem dari sudut pandang pengguna, sehingga dapat menjelaskan apa saja yang dilakukan oleh sistem tanpa perlu menjelaskan bagaimana sistem tersebut bekerja secara teknis. Simbol-simbol yang digunakan dalam Use Case Diagram disajikan secara sistematis dalam Tabel 2.2 untuk memudahkan pemahaman mengenai fungsi dan peran masing-masing simbol dalam pemodelan sistem.

Tabel 2. 2 Use Case Diagram

No	Simbol	Nama	Keterangan
1.		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case.*
2.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(Independent)* akan mempengaruhi elemen yang bergantung pada elemen yang tidak mandiri.
3.		*Generalization*	Hubungan dimana objek anak *(descedent)* perliaku dan struktur data dari objek yang ada di atasnya objek induk *(ancestor).*

4.		*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit.
5.		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang di berikan.
6.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.		*Use case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang

			terukur bagi suatu aktor.
9.		*Collaboration*	Interaksi aturqan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10.		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Sumber: diadaptasi dari berbagai referensi notasi UML

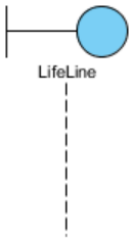
2. Sequence Diagram

Untuk menggambarkan urutan antara objek-objek dalam sistem secara kronologis, diperlukan sebuah diagram yang mampu menunjukan bagaimana pesan atau informasi dikirimkan antara satu objek dengan objek lainnya dalam rentang waktu

tertentu. Diagram ini sangat bermanfaat dalam memodelkan alur komunikasi antar komponen sistem, terutama ketika terdapat proses yang kompleks dan melibatkan banyak objek sekaligus. Simbol-simbol yang digunakan dalam sequence disajikan secara sistematis dalam Tabel 2.3 untuk memudahkan pemahaman mengenai fungsi dan peran masing-masing simbol dalam menggambarkan urutan interaksi antar objek dalam sistem.

Tabel 2. 3 Sequence Diagram

No	Simbol	Nama	Keterangan
1.		*Actor*	Pengguna atau pihak luar yang melakukan interaksi dengan sistem.
2.		*Message*	Komunikasi antar objek yang menunjukan proses berjalan.
3.		*Reply Message*	Komunikasi balasan antara objek sebagai respon proses.

4.		*Boundary Lifeline*	Bagian sistem yang menangani interaksi antara aktor dengan sistem.
5.		*Control Lifeline*	Bagian sistem yang mengontrol alur proses dan logika sistem.
6.		*Entity Lifeline*	Bagian sistem yang mempresentasikan data atau basis data.

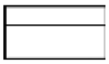
Sumber: diadaptasi dari berbagai referensi notasi UML

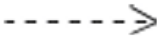
3. Class Diagram

Dalam perancangan sistem berbasis objek, dibutuhkan sebuah diagram yang mampu menggambarkan struktur statis sistem dengan menampilkan kelas-kelas yang ada beserta atribut, operasi, dan hubungan antar kelas. Diagram ini juga berperan penting sebagai dasar perancangan basis data, karena kelas-kelas yang ada dapat dipetakan langsung menjadi tabel dalam basis data realsional. Simbol-simbol yang digunakan dalam Class Diagram dalam Tabel 2.4 untuk memudahkan pemahaman mengenai

fungsi dan peran masing-masing simbol dalam menggambarkan struktur dan hubungan antarkelas dalam sistem.

Tabel 2. 4 Class Diagram

No	Simbol	Nama	Keterangan
1.		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi. perilaku dan struktur data dari objek yang ada di atasnya objek induk *(ancestor).*
2.		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3.		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4.		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem

			yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5.		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6.		*Dependecy*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(Independent)* akan mempengaruhi elemen yang tidak mandiri.
7.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

Sumber: diadaptasi dari berbagai referensi notasi UML

4. Activity Diagram

Untuk memodelkan alur kerja atau proses bisnis dalam sistem secara berurutan, diperlukan sebuah diagram yang

menggambarkan rangkaian aktivitas dari awal hingga akhir suatu proses, termasuk percabangan kondisi dan aktivitas yang berjalan secara paralel. Diagram ini sangat membantu dalam memvisualisasikan proses yang terjadi di balik setiap fitur sistem sehingga memudahkan pemahaman bagi pengembang maupun pemangku kepentingan. Simbol-simbol yang digunakan Activity Diagram disajikan secara sistematis dalam Tabel 2.5 untuk memudahkan pemahaman mengenai fungsi dan peran masing-masing simbol dalam memodelkan alur kerja dan proses bisnis sistem.

Tabel 2. 5 Activity Diagram

No	Simbol	Nama	Keterangan
1.		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
2.		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
3.		*Initial Node*	Bagaimana objek dibentuk atau di awali.

4.		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan.
5.		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.

Sumber: diadaptasi dari berbagai referensi notasi UML

2.2.2 ESP 32

Arsitektur sistem monitoring berbasis IoT memerlukan mikrokontroler yang memiliki kemampuan komputasi dan konektivitas jaringan terintegrasi. Salah satu perangkat yang banyak digunakan adalah *ESP32*, yaitu mikrokontroler *System-on-a-Chip (SoC)* yang dikembangkan oleh Espressif Systems dan dirancang untuk mendukung berbagai aplikasi IoT [35]. Mikrokontroler ini memiliki prosesor *dual-core* serta telah terintegrasi dengan modul *Wi-Fi* dan *Bluetooth Low Energy* (*BLE*) [36]. ESP32 mampu bekerja pada tegangan 3,3V dan berfungsi sebagai pengolah utama data kebisingan dan getaran yang diterima dari sensor, kemudian mengirimkan data tersebut secara *real-time* ke *server* berbasis *web* melalui koneksi *Wi-Fi* [37]. Kemampuan pemrosesan yang cepat dan dukungan komunikasi nirkabel menjadikan ESP32 sangat sesuai digunakan pada sistem *monitoring real-time* berbasis *IoT* [38].

2.2.3 Firebase

Salah satu platform komputasi awan berbasi NoSQL yang dikembangkan oleh Google untuk memfasilitasi manajemen data secara efisien melalui format JavaScript Object Notation (JSON) adalah firebase [39], [40]. Keunggulan utamanya terletak pada fitur realtime database yang memungkinkan sinkronisasi data secara instan dengan latensi rendah antara perangkat IoT dan antarmuka pengguna [41]. Selain itu, firebase menyediakan infrastruktur keamanan terintegrasi melalui mekanisme autentifikasi dan otorisasi guna menjamin kerahasiaan serta integritas data yang di transmisikan [41].

2.2.4 Laravel

Dalam pengembangan aplikasi web modern. Terdapat sebuah kerangka kerja PHP berbasis open-source yang mengintegrasikan arsitektur Model View Controller (MVC) untuk menghasilkan aplikasi yang terstruktur, modular, dan skalabel [42], [43] , yaitu Laravel. Platform ini dikenal memiliki sintaks yang elegen serta dilengkapi fitur keamanan bawaan, seperti enkripsi dan validasi, yang mengoptimalkan manajemen data serta otentifikasi pengguna [44], [45]. Dalam ekosistem *IoT*, Laravel berfungi sebagai penyedia *API* yang andai API yang andal untuk mengelola transmisi dan visualisasi data senso secara real-time [46], [47].

2.2.5 MySQL

Sistem Manajemen Basis Data Relasional (RDMBS) berbasis open-source yang mengimplementasikan *Structured Query Language* (SQL) untuk mengelola penyimpanan serta pengambilan dalam skala besar secara efisien dikenal dengan nama MySQL [48], [49]. Sistem ini memiliki performa unggul dalam menangani transaksi berkecepatan tinggi, khususnya pada lingkungan yang didominasi operasi pembacaan data, sehingga menjadi standar utama bagi arsitektur aplikasi bebasis *web* [50]. Dalam integrasi IoT, MySQL berfungsi sebagai repositori data historis yang menyimpan informasi dari sensor secara terstruktur guna memfasilitasi analisis data dan visualisasi pada antarmuka pemantuan [51] [52].

2.2.6 MQTT

Protokol pesan standar berbasis arsitektur publish-subscribe yang dirancang untuk konektivitas perangkat jarak jauh dengan jejak kode minimal dan penggunaan bandwidth yang sangat efisien dikenal sebagai MQTT *(Message Telemetry Transport)* [46], [53]. Protokol ini sangat optimal bagi ekosistem *IoT* karena sifatnya yang ringan, hemat daya, serta mampu memfasilitasi komunikasi mesin-ke-mesin (M2M) secara asinkron melalui peran perantara (broker) [54], [55]. Selain itu, MQTT mendukung fitur *Qualitiy of Service* (QoS) dalam tiga tingkatan untuk menjamin keandalan serta integritas transmisi data pada kondisi jaringan yang tidak stabil [18], [56].

2.2.7 ISO 1996-1:2016

Standar internasional yang mengatur besaran dasar dan prosedur penilaian kebisingan lingkungan dikenal sebagai ISO 1996-1:2016. Standar ini mendefinisikan deskriptor kebisingan berupa *Equivalent Continuous Sound Pressure Level* (LAeq), yaitu representasi tingkat tekanan bunyi rata-rata dalam satuan dB(A) pada suatu interval waktu pengukuran tertentu. Selain itu, standar ini juga mengatur interval waktu acuan *(reference time interval)* yang menjadi dasar dalam karakterisasi kebisingan lingkungan, sehingga nilai LAeq yang dihasilkan sistem dapat dibandingkan secara konsisten terhadap Nilai Ambang Batas (NAB) sebesar 85 dB(A).

2.2.8 ISO 2631-1:1997

Standar internasional yang mengatur metode evaluasi paparan getaran seluruh tubuh *(whole-body vibration)* terhadap manusia dikenal sebagai ISO 2631-1:1997. Standar ini mengatur metode pengukuran percepatan getaran pada tiga sumbu (x, y, z) yang kemudian diolah menjadi nilai root mean square (rms) dalam satuan m/s^2. Nilai tersebut digunakan untuk mengategorikan tingkat ketidaknyamanan akibat getaran, mulai dari kategori tidak terasa hingga sangat tidak nyaman, serta menjadi acuan dalam penetapan Nilai Ambang Batas (NAB) getaran sebesar 0,5 m/s^2 pada sistem yang dikembangkan.