RANCANG BANGUN SISTEM PEMANTAUAN SIMULASI KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*

TUGAS AKHIR

Oleh:

ANDI THEDA MUFARRIHAH ANWAR
NIT.30423002

PROGRAM STUDI TEKNIK PESAWAT UDARA
POLITEKNIK PENERBANGAN SURABAYA
2026

RANCANG BANGUN SISTEM PEMANTAUAN SIMULASI KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat untuk Mendapatkan Gelar Ahli Madya (A.Md)
pada Program Studi Diploma 3 Teknik Pesawat Udara

Oleh:

ANDI THEDA MUFARRIHAH ANWAR
NIT.30423002

PROGRAM STUDI TEKNIK PESAWAT UDARA
POLITEKNIK PENERBANGAN SURABAYA
2026

HALAMAN PERSETUJUAN

RANCANG BANGUN SISTEM PEMANTAUAN SIMULASI KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*

Oleh :

ANDI THEDA MUFARRIHAH ANWAR
NIT. 30423002

Disetujui untuk diujikan :
Surabaya, 16 Juli 2026

Pembimbing I : Dr. SUYATMO. S.Pd, S.T.,M.T.
NIP. 19630510 198902 1 001

Pembimbing II : YUDHIS THIRO KABUL. M.Kom.
NIP. 19870224 202203 1 003

HALAMAN PENGESAHAN

RANCANG BANGUN SISTEM PEMANTAUAN SIMULASI KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*

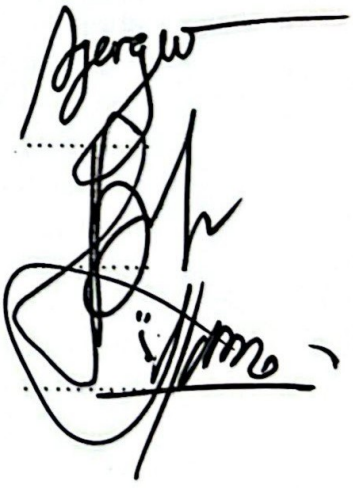
Oleh :

ANDI THEDA MUFARRIHAH ANWAR
NIT.30423002

Telah dipertahankan dan dinyatakan lulus pada Proyek Akhir
Program Pendidikan Diploma 3 Teknik Pesawat Udara
Politeknik Penerbangan Surabaya
Pada tanggal : 16 Juli 2026

Panitia Penguji :

1. Ketua : AJENG WULANSARI, ST, MT
   NIP. 19890606 200912 2 001
2. Sekretaris : Dr. BAMBANG BAGUS H., S.SiT, MM, MT
   NIP. 19810915 200502 1 001
3. Anggota : Dr. SUYATMO. S.Pd, S.T.,M.T
   NIP. 19630510 198902 1 001



Ketua Program Studi

D3 TEKNIK PESAWAT UDARA

NYARIS PAMBUDIYATNO, S.SiT, M.MTr
NIP. 19820515 200502 1 001

PERNYATAAN KEASLIAN DAN HAK CIPTA

Saya yang bertanda tangan di bawah ini :

Nama : Andi Theda Mufarrihah
NIT 30423002
Program Studi : D3 Teknik Pesawat Udara
Judul Proyek Akhir/Tugas Akhir : Rancang Bangun Sistem Pemantauan Simulasi Kebisingan Pesawat Udara Berbasis IOT dengan Visualisasi *Real-Time*

dengan ini menyatakan bahwa :

1. Proyek Akhir/Tugas Akhir ini merupakan karya asli dan belum pernah diajukan untuk mendapatkan gelar akademik, baik di Politeknik Penerbangan Surabaya maupun di Perguruan Tinggi lain, serta dipublikasikan, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.

2. Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan Hak Bebas Royalti Non Eksklusif (*Non-Exclusive Royalty-Free Right*) kepada Politeknik Penerbangan Surabaya beserta perangkat yang ada (jika diperlukan). Dengan hak ini, Politeknik Penerbangan Surabaya berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan Proyek Akhir/Tugas Akhir saya dengan tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya. Apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi dan Akademi Penerbangan.

Surabaya, 16 Juli 2026

Andi Theda Mufarrihah
NIT. 30423002

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa, karena berkat limpahan rahmat dan karunia-Nya, Tugas Akhir yang berjudul **RANCANG BANGUN SIMULASI SISTEM PEMANTAUAN KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*** Ini dapat diselesaikan dengan baik. Penyusunan Proyek Akhir ini dimaksudkan sebagai salah satu syarat menyelesaikan pendidikan di Politeknik Penerbangan Surabaya dan memperoleh gelar Ahli Madya (A.Md.).

Ucapan terima kasih kami sampaikan kepada segenap pihak yang telah membantu selama proses penyusunan Proyek Akhir ini, terutama kepada :

1. Bapak Ahmad Bahrawi, S.E., M.T.. selaku Direktur Politeknik Penerbangan Surabaya.
2. Bapak Nyaris Pambudiyatno, S.SiT., M.MTr., selaku Ketua Program Studi D3 Teknik Pesawat Udara di Politeknik Penerbangan Surabaya
3. Bapak Dr. Suyatmo, ST , S.Pd, MT. selaku dosen pembimbing I, atas bimbingannya dalam materi yang senantiasa membimbing dan membantu dalam penyusunan naskah proyek akhir.
4. Bapak Yudhis Thiro Kabul Yunior, M.Kom selaku Dosen Pembimbing II, atas bimbingannya dalam penulisan naskah proyek akhir.
5. Seluruh dosen dan civitas akademika Program Studi Teknik Pesawat Udara Politeknik Penerbangan Surabaya yang telah memberikan ilmu yang bermanfaat.
6. Kepada Bapak Drs. Andi Anwar (Alm) dan Ibu Aminah selaku orang tua saya beserta keluarga tercinta yang senantiasa memberikan doa serta bantuan secara materi dan dukungan moral untuk kelancaran proyek akhir ini.
7. Seluruh rekan-rekan D3 Teknik Pesawat Udara angkatan IX yang selalu memberikan bantuan dukungan, dan motivasi selama menempuh Pendidikan di Politeknik Penerbangan Surabaya.
8. Segenap sahabat, senior, junior, dan semua pihak yang tidak dapat disebutkan satu per satu, namun telah memberikan bantuan, inspirasi, dan dukungan selama proses penelitian ini berlangsung.

Penulis menyadari bahwa karya ini masih jauh dari sempurna. Namun setiap kekurangan adalah ruang untuk bertumbuh, dan setiap kritik maupun saran yang membangun akan menjadi cahaya untuk perjalanan akademik selanjutnya. Semoga proyek akhir ini tidak hanya memberi manfaat bagi dunia pendidikan, tetapi juga menjadi langkah kecil menuju masa depan teknologi pemantauan kebisingan yang lebih baik.

Surabaya, 16 Juni 2026

Andi Theda Mufarrihah Anwar

ABSTRAK

RANCANG BANGUN SISTEM PEMANTAUAN SIMULASI KEBISINGAN PESAWAT UDARA BERBASIS IOT DENGAN VISUALISASI *REAL-TIME*

Oleh:
Andi Theda Mufarrihah
NIT : 30423002

Kebisingan pesawat udara merupakan salah satu isu lingkungan yang berdampak terhadap kenyamanan masyarakat, kesehatan, serta keselamatan kerja di sekitar bandar udara. Sistem pemantauan kebisingan yang digunakan saat ini umumnya masih bersifat konvensional, tidak terintegrasi dengan jaringan, serta memerlukan proses pengambilan dan pengolahan data secara manual. Penelitian ini bertujuan untuk merancang dan membangun prototipe sistem pemantauan simulasi kebisingan pesawat udara berbasis *Internet of Things* (IoT) dengan kemampuan visualisasi data secara *real-time.* Sistem dikembangkan menggunakan sensor kebisingan HH_06.03 (FaTHOR) yang berkomunikasi melalui antarmuka Serial TTL (UART), mikrokontroler ESP32, komunikasi Wi-Fi, database MySQL, dan dashboard web sebagai media monitoring. Data kebisingan dalam satuan dB(A) diperoleh dari sensor, kemudian dikirim secara otomatis ke server dan ditampilkan dalam bentuk grafik serta informasi numerik secara real-time. Pengujian sistem dilakukan pada lingkungan laboratorium menggunakan speaker sebagai sumber simulasi kebisingan pesawat udara. Evaluasi sistem meliputi pengujian akurasi sensor terhadap *Sound Level Meter* (SLM) sebagai alat referensi, pengujian stabilitas koneksi data, serta pengujian *latency* pengiriman data menuju *dashboard*. Hasil pengujian menunjukkan bahwa sistem mampu menampilkan data kebisingan secara *real-time* melalui dashboard web dengan keberhasilan pengiriman data yang baik. Pengujian akurasi menggunakan *Sound Level Meter* (SLM) sebagai alat referensi menunjukkan bahwa sensor HH_06.03 memiliki nilai error rata-rata yang rendah sehingga layak digunakan untuk pemantauan kebisingan. Dengan demikian, sistem yang dirancang mampu melakukan pemantauan simulasi kebisingan pesawat udara secara berkelanjutan dan terintegrasi melalui teknologi *Internet of Things* (IoT).

Kata Kunci: Kebisingan pesawat, IoT, ESP32, sensor mikrofon, monitoring *real-time*, visualisasi data.

ABSTRACT

DESIGN AND DEVELOPMENT OF AN IOT-BASED AIRCRAFT NOISE MONITORING SYSTEM WITH REAL-TIME VISUALIZATION

By:
Andi Theda Mufarrihah
NIT : 30423002

Aircraft noise is one of the environmental issues that affects public comfort, health, and occupational safety in areas surrounding airports. Existing noise monitoring systems are generally conventional, not integrated with network-based platforms, and require manual data collection and processing. This study aims to design and develop a prototype of an Internet of Things (IoT)-based aircraft noise simulation monitoring system with real-time data visualization capabilities. The system was developed using an HH_06.03 (FaTHOR) noise sensor communicating through a Serial TTL (UART) interface, an ESP32 microcontroller, Wi-Fi communication, a MySQL database, and a web-based dashboard for monitoring purposes. Noise level data in dB(A) are acquired by the sensor, automatically transmitted to the server, and displayed in real-time through graphical and numerical visualizations. System testing was conducted in a laboratory environment using a speaker as a simulated aircraft noise source. The evaluation focused on sensor accuracy compared to a Sound Level Meter (SLM) as the reference instrument, data transmission stability, and latency in delivering data to the monitoring dashboard. Test results show that the system is capable of displaying noise data in real time via a web dashboard with a high data transmission success rate. Accuracy testing using a Sound Level Meter (SLM) as a reference tool showed that the HH_06.03 sensor has a low average error value, making it suitable for noise monitoring. Thus, the designed system is capable of continuously and integrally monitoring simulated aircraft noise through Internet of Things (IoT) technology.

Keywords*: Aircraft noise, IoT, ESP32, microphone sensor, real-time monitoring, data visualization.*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR PUSTAKA

Adfry, D. P., Muskhir, M., & Luthfi, A. (2023). Efektivitas Pembelajaran Pemograman Mikrokontroler Menggunakan Aplikasi Arduino IDE. *Jurnal Pendidikan Teknik Elektro*, *4*(2), 321–329. https://doi.org/10.24036/jpte.v4i2.312

Ahuja, V., Little, D. S., Majdalani, J., & Hartfield, R. J. (2022). On the prediction of noise generated by urban air mobility (UAM) vehicles. Part II. Implementation of the Farassat F1A formulation into a modern surface-vorticity panel solver. *Physics of Fluids*, *34*(11). https://doi.org/10.1063/5.0105002

Babiuch, M., & Postulka, J. (2021). Smart Home Monitoring System Using ESP32 Microcontrollers. *Internet of Things*, 1–21. https://doi.org/10.5772/intechopen.94589

Bą czalska, J., Wojciechowska, W., Rojek, M., Hahad, O., Daiber, A., Münzel, T., & Rajzer, M. (2022). Cardiovascular consequences of aircraft noise exposure. *Frontiers in Public Health*, *10*. https://doi.org/10.3389/fpubh.2022.1058423

Bakar, Z. A., Zairil, M., Nor, M., Kadiran, K. A., Misnan, M. F., & Noorezam, M. (2022). *Smart Plant Monitoring System Using Aquaponics Production Technological with Arduino Development Environment (IDE) and SMS Alert : A Prototype*. 32–47.

Borkovkina, S., Camino, A., Janpongsri, W., Sarunic, M. V., & Jian, Y. (2020). Real-time retinal layer segmentation of OCT volumes with GPU accelerated inferencing using a compressed, low-latency neural network. *Biomedical Optics Express*, *11*(7), 3968. https://doi.org/10.1364/boe.395279

Fatema, T., Hakim, M. A., Mim, T. K., Mitu, M. J., & Paul, B. (2023). IoT cloud based noise intensity monitoring system. *Indonesian Journal of Electrical Engineering and Computer Science*, *30*(1), 289–298. https://doi.org/10.11591/ijeecs.v30.i1.pp289-298

Giladi, R. (2020). *Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID- 19 . The COVID-19 resource centre is hosted on Elsevier Connect , the company ' s public news and information* . (January).

Gyorodi, C., Dumse-Burescu, D., Zmaranda, D., Gyorodi, R., Gobor, G., & Pecherle, G. (2020). Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application's Data Storage. *Appl. Sci.*

International Civil Aviation Organization. (2025). *Report on Good Practices of Noise Monitoring Systems Deliverable*.

Jin, Y. H., Hwang, I. T., & Lee, W. H. (2020). A mobile augmented reality system for the real-time visualization of pipes in point cloud data with a depth sensor. *Electronics (Switzerland)*, *9*(5). https://doi.org/10.3390/electronics9050836

Khan, A. U., Khan, M. E., Hasan, M., Zakri, W., Alhazmi, W., & Islam, T. (2022). An Efficient Wireless Sensor Network Based on the ESP-MESH Protocol for Indoor

and Outdoor Air Quality Monitoring. *Sustainability (Switzerland)*, *14*(24). https://doi.org/10.3390/su142416630

Khan, M. A., Din, I. U., Kim, B. S., & Almogren, A. (2023). Visualization of Remote Patient Monitoring System Based on Internet of Medical Things. *Sustainability (Switzerland)*, *15*(10), 1–15. https://doi.org/10.3390/su15108120

Kusumah, H., & Pradana, R. A. (2019). Penerapan Trainer Interfacing Mikrokontroler Dan Internet of Things Berbasis Esp32 Pada Mata Kuliah Interfacing. *Journal CERITA*, *5*(2), 120–134. https://doi.org/10.33050/cerita.v5i2.237

Lata, K., Sood, A., Kaur, K., Benipal, A. K., & Pateriya, B. (2022). Web-GIS based Dashboard for Real-Time Data Visualization & Analysis using Open Source Technologies. *Journal of Geomatics*, *16*(2), 134–146. https://doi.org/10.58825/jog.2022.16.2.42

Liu, J., Sun, S., Tang, K., Fan, X., Lv, J., Fu, Y., Feng, X., & Zeng, L. (2025). IoT-Based Airport Noise Perception and Monitoring: Multi-Source Data Fusion, Spatial Distribution Modeling, and Analysis. *Sensors*, *25*(8), 1–29. https://doi.org/10.3390/s25082347

Lyashenko, V., Abu-Jassar, A. T., Yevsieiev, V., & Maksymova, S. (2023). Automated Monitoring and Visualization System in Production. *International Research Journal of Multidisciplinary Technovation*, *5*(6), 09–18. https://doi.org/10.54392/irjmt2362

Mutmainnah, N., Lapian, F. E. P., Teknik, F., Yapis, U., & Sentani, H. E. (2025). *ANALISIS KEBISINGAN BANDAR UDARA DORTHEYS HIYO ELUAY SENTANI JAYAPURA AKIBAT AKTIVITAS PESAWAT*. 144–152.

Nizam, M. N., Haris Yuana, & Zunita Wulansari. (2022). Mikrokontroler Esp 32 Sebagai Alat Monitoring Pintu Berbasis Web. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *6*(2), 767–772. https://doi.org/10.36040/jati.v6i2.5713

Nöding, M., Schuermann, M., Bertsch, L., Koch, M., Plohr, M., Jaron, R., & Berton, J. J. (2022). Simulation of landing and take-off noise for supersonic transport aircraft at a conceptual design fidelity level. *Aerospace*, *9*(1), 1–23. https://doi.org/10.3390/aerospace9010009

Plauska, I., Liutkevičius, A., & Janavičiūtė, A. (2023). Performance Evaluation of C/C++, MicroPython, Rust and TinyGo Programming Languages on ESP32 Microcontroller. *Electronics (Switzerland)*, *12*(1). https://doi.org/10.3390/electronics12010143

Rahanjani, Y. E., & Nugraha, B. (2020). Real-Time Data Transmission and Visualization as a Powerful Technology to Reduce Non-Productive Time During Drilling Operations: Present Day Capabilities, Limitation, and Future Development. *Scientific Contributions Oil and Gas*, *43*(3), 135–142. https://doi.org/10.29017/SCOG.43.3.515

Ramadhan, N. P., Fachrul, M. F., & Widyatmoko, W. (2019). Kawasan Kebisingan Bandar Udara Internasional Husein Sastranegara, Bandung, Provinsi Jawa Barat.

Journal of Environmental Engineering and Waste Management, *4*(2), 43. https://doi.org/10.33021/jenv.v4i2.767

Rifqah, R. A. N., Suciyati, S. W., Surtono, A., & Pauzi, G. A. (2023). *Design of a Classroom Noise Monitoring Tool Using a KY-037 Sound Sensor Based on Wemos D1R1*. *4*(4).

Rijnders, F., Wallner, G., & Bernhaupt, R. (2022). Live Feedback for Training Through Real-Time Data Visualizations: A Study with League of Legends. *Proceedings of the ACM on Human-Computer Interaction*, *6*(2022). https://doi.org/10.1145/3549506

Rizdqi Akbar Ramadhan, Abdul Kudus Zaini, & Bima Kristian Pranoto. (2022). Edukasi Pemrograman WEB Fundamental Sebagai Ilmu Wajib Era Industri 4.0. *Jurnal Pengabdian Masyarakat Dan Penerapan Ilmu Pengetahuan*, *3*(1), 11–15. https://doi.org/10.25299/jpmpip.2022.10591

Rötger, T., Eyers, C., & Fusaro, R. (2024). A Review of the Current Regulatory Framework for Supersonic Civil Aircraft: Noise and Emissions Regulations. *Aerospace*, *11*(1), 1–23. https://doi.org/10.3390/aerospace11010019

Saldana-Barrios, J. J., Aguilar, E., Ng, W., & Orocu, R. (2023). Designing an IoT-Based System for Monitoring Noise Levels in the Computer Science Faculty and Library of the Technological University of Panama. *Sensors*, *23*(22). https://doi.org/10.3390/s23229083

Salunke, S. V., & Ouda, A. (2024). A Performance Benchmark for the PostgreSQL and MySQL Databases. *Future Internet*, *16*(10). https://doi.org/10.3390/fi16100382

Simons, D. G., Besnea, I., Mohammadloo, T. H., Melkert, J. A., & Snellen, M. (2022). Comparative assessment of measured and modelled aircraft noise around Amsterdam Airport Schiphol. *Transportation Research Part D: Transport and Environment*, *105*(January), 103216. https://doi.org/10.1016/j.trd.2022.103216

Sinlae, F., Maulana, I., Setiyansyah, F., & Ihsan, M. (2024). Pengenalan Pemrograman Web: Pembuatan Aplikasi Web Sederhana Dengan PHP dan MYSQL. *Jurnal Siber Multi Disiplin*, *2*(2), 68–82. https://doi.org/10.38035/jsmd.v2i2.156

Toni, M., & Sudin, M. (2024). *Research and Development (R & D) Interactive Media That Is Effective , Efficient and Fun for Students*. *2*, 1239–1252.

World Health Organization, W. (2018). Who environmental noise guidelines for the european region. *Larmbekampfung*, *13*(6), 200–203.

Zenhadi. (2022). PENGENALAN ESP32 BOARD. In *MK Internet of Things* (Vol. 6). https://zenhadi.lecturer.pens.ac.id/kuliah/InternetOfThings/Praktek/Praktek Modul 1 Pengenalan ESP32.pdf

Zhang, X., Zhao, M., & Dong, R. (2020). Time-series prediction of environmental noise for urban iot based on long short-term memory recurrent neural network. *Applied Sciences (Switzerland)*, *10*(3). https://doi.org/10.3390/app10031144

Zmaranda, D. R., Moisi, C. I., Győrödi, C. A., Győrödi, R., & Bandici, L. (2021). An analysis of the performance and configuration features of mysql document store

and elasticsearch as an alternative backend in a data replication solution. *Applied Sciences (Switzerland)*, *11*(24). https://doi.org/10.3390/app112411590

LAMPIRAN

Lampiran 1 Program *(Source kode)* ESP32

```
//
// ═══════════════════════════════════════════════════════════════
//  ANMS – Aircraft Noise Monitoring System
//  ALAT 1 – Sensor 1
//  ESP32 → HTTP POST → Node.js API → MySQL XAMPP
//
//  Library yang dibutuhkan (install via Library Manager):
//    • Arduino ESP32 core  (by Espressif Systems)
//    • ArduinoJson         (by Benoit Blanchon, versi 6.x)
//
//  TIDAK membutuhkan Blynk library lagi.
//
// ═══════════════════════════════════════════════════════════════

#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <time.h>
#include <math.h>

//
// ═══════════════════════════════════════════════════════════════
//  KONFIGURASI – SESUAIKAN BAGIAN INI
//
// ═══════════════════════════════════════════════════════════════

// ── WiFi ──────────────────────────────────────────────────────
const char* WIFI_SSID     = "ThedaTA";
const char* WIFI_PASSWORD = "nutellaaa";

// ── Server API ────────────────────────────────────────────────
// Ganti dengan IP komputer yang menjalankan XAMPP + Node.js
// Cara cek IP: ketik "ipconfig" di CMD Windows → IPv4 Address
const char* SERVER_IP     = "192.168.100.124";  // ← GANTI INI
const int   SERVER_PORT   = 3000;
const char* API_ENDPOINT  = "/api/data";
```

```
// ── Keamanan ──────────────────────────────────────────────────────────
// Harus sama persis dengan DEVICE_API_KEY di file backend/.env
const char* API_KEY      = "MASUKKAN_SESUAI_TULISAN_INI"; // ← GANTI INI

// ── Identitas alat ini ────────────────────────────────────────
const char* DEVICE_ID    = "NodeMCU-01";  // Sensor 1
const char* SENSOR_LABEL  = "Sensor 1";

// ── Pin & ADC ─────────────────────────────────────────────────
#define MIC_PIN          34
const int   NUM_SAMPLES   = 1000;

// ── Interval kirim data (milidetik) ──────────────────────────
// Default: 60000 = 1 menit (sesuai kebutuhan pencatatan)
// Ubah ke 5000 untuk testing (5 detik)
const unsigned long INTERVAL_MS = 60000UL;

// ── NTP / Waktu ──────────────────────────────────────────────
const char* NTP_SERVER      = "pool.ntp.org";
const long  GMT_OFFSET_SEC  = 7 * 3600;  // WIB (UTC+7)
const int  DAYLIGHT_OFFSET  = 0;

//
// ═══════════════════════════════════════════════════════════════
// VARIABEL GLOBAL
//
// ═══════════════════════════════════════════════════════════════

float offsetADC    = 0;
float rms          = 0;
float vrms          = 0;
float dB           = 0;
String jamSekarang  = " -------";
String tanggalSekarang = "--/--/ --- ";
String jenisPesawat = "-";
String statusPesawat = "Idle";
```

```
unsigned long lastSent = 0;
bool  wifiOK = false;

//
// ================================================================
//  UPDATE TANGGAL & JAM DARI NTP
//
// ================================================================

void updateDateTime()
{
  struct tm timeinfo;
  if (!getLocalTime(&timeinfo)) {
    jamSekarang     = "-------";
    tanggalSekarang = "--/--/ ---";
    return;
  }

  char jamBuf[10];
  strftime(jamBuf, sizeof(jamBuf), "%H:%M:%S", &timeinfo);
  jamSekarang = String(jamBuf);

  char tglBuf[20];
  strftime(tglBuf, sizeof(tglBuf), "%d/%m/%Y", &timeinfo);
  tanggalSekarang = String(tglBuf);
}

//
// ================================================================
//  HITUNG NOISE (RMS → VRMS → dB)
//  Logika identik dengan program asli
//
// ================================================================

void hitungNoise()
{
  // Langkah 1: hitung offset DC ADC
  long total = 0;
  for (int i = 0; i < NUM_SAMPLES; i++) {
    total += analogRead(MIC_PIN);
    delayMicroseconds(100);
```

```
  }
  offsetADC = (float)total / NUM_SAMPLES;

  // Langkah 2: hitung RMS sinyal AC
  double sumSquare = 0;
  for (int i = 0; i < NUM_SAMPLES; i++) {
    float sample  = analogRead(MIC_PIN);
    float centered = sample - offsetADC;
    sumSquare     += centered * centered;
    delayMicroseconds(100);
  }
  rms = sqrt(sumSquare / NUM_SAMPLES);

  // Langkah 3: konversi ke tegangan RMS (VRMS)
  vrms = rms * 3.3 / 4095.0;
  if (vrms < 0.000001) vrms = 0.000001;

  // Langkah 4: konversi ke dB dengan rumus kalibrasi
  dB = 27.55 * log10(vrms) + 96.90;
  if (dB < 0) dB = 0;
  dB = round(dB * 10.0) / 10.0;
}

//
==========================================================
// PERKIRAAN JENIS PESAWAT BERDASARKAN RENTANG dB
//  Logika identik dengan program asli
//
==========================================================

String getPesawat(float db)
{
  if      (db < 55)  return "Normal";
  else if (db < 75)  return "Wings Air";
  else if (db < 85)  return "Batik Air";
  else if (db < 87)  return "Citilink";
  else if (db < 95)  return "Lion Air";
  else               return "Garuda Indonesia";
}

//
==========================================================
```

```
//  STATUS TAKEOFF / LANDING / IDLE
//  Disesuaikan dengan kolom ENUM MySQL: Takeoff, Landing, Idle
//  - dB >= 70 dan naik → Takeoff
//  - dB >= 70 dan turun → Landing
//  - dB < 70          → Idle
//
//============================================================

float dB_sebelumnya = 0;

String getStatus(float db)
{
  String st;
  if (db < 70) {
    st = "Idle";
  } else if (db >= dB_sebelumnya) {
    st = "Takeoff";
  } else {
    st = "Landing";
  }
  dB_sebelumnya = db;
  return st;
}

//
//============================================================
//  KONEKSI WIFI
//
//============================================================

void koneksiWiFi()
{
  Serial.print("[WiFi] Menghubungkan ke ");
  Serial.println(WIFI_SSID);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  int coba = 0;
  while (WiFi.status() != WL_CONNECTED && coba < 20) {
    delay(500);
    Serial.print(".");
    coba++;
  }
```

```
  if (WiFi.status() == WL_CONNECTED) {
    wifiOK = true;
    Serial.println();
    Serial.print("[WiFi] Terhubung! IP ESP32: ");
    Serial.println(WiFi.localIP());
  } else {
    wifiOK = false;
    Serial.println();
    Serial.println("[WiFi] GAGAL terhubung. Coba lagi nanti...");
  }
}

//
==========================================================
// KIRIM DATA KE SERVER VIA HTTP POST
//
==========================================================

void kirimKeServer()
{
  // Pastikan WiFi masih terhubung
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("[HTTP] WiFi terputus, reconnect...");
    koneksiWiFi();
    return;
  }

  // Bangun URL
  String url = "http://";
  url += SERVER_IP;
  url += ":";
  url += SERVER_PORT;
  url += API_ENDPOINT;

  // Bangun JSON payload
  // Field harus cocok dengan tabel sensor_data MySQL:
  //  sensor1_db, jenis_pesawat, status, device_id
  StaticJsonDocument<256> doc;
  doc["sensor1_db"]    = dB;         // ← field MySQL sensor1_db
  doc["sensor2_db"]    = 0;          // Alat 1 hanya punya 1 sensor,
  doc["sensor3_db"]    = 0;          //  isi semua dengan nilai yang sama
```

```
                                //   atau 0 jika ingin dibedakan
  doc["jenis_pesawat"] = jenisPesawat; // ← field MySQL jenis_pesawat
  doc["status"]        = statusPesawat;// ← field MySQL status (Takeoff/Landing/Idle)
  doc["device_id"]     = DEVICE_ID;  // ← field MySQL device_id

  String payload;
serializeJson(doc, payload);
WiFiClient test;

Serial.print("Tes koneksi ke server... ");

if (test.connect(SERVER_IP, SERVER_PORT)) {
    Serial.println("BERHASIL");
    test.stop();
} else {
    Serial.println("GAGAL");
}
  // Kirim HTTP POST
  WiFiClient client;
HTTPClient http;

http.begin(client, url);

Serial.print("Tes koneksi ke server... ");
if (client.connect(IPAddress(10,164,228,13), 3000)) {
  Serial.println("BERHASIL");
  client.stop();
} else {
  Serial.println("GAGAL");
}

http.begin(client, url);
http.addHeader("Content-Type", "application/json");
http.addHeader("x-api-key", API_KEY);
http.setTimeout(8000);  // timeout 8 detik

  int httpCode = http.POST(payload);

  if (httpCode > 0) {
    Serial.print("[HTTP] Response: ");
    Serial.print(httpCode);
    if (httpCode == 201) {
```

```
      Serial.println("  Data tersimpan ke MySQL");
    } else {
      Serial.print(" | Body: ");
      Serial.println(http.getString());
    }
  } else {
    Serial.print("[HTTP] ERROR: ");
    Serial.println(http.errorToString(httpCode));
    Serial.println("       Pastikan server Node.js (START_SERVER.bat) sedang
berjalan");
  }

  http.end();
}

//
============================================================
//  SERIAL MONITOR – TAMPILKAN DATA
//
============================================================

void tampilkanSerial()
{
  Serial.println("==========================================");
  Serial.print("Jam       : "); Serial.println(jamSekarang);
  Serial.print("Tanggal   : "); Serial.println(tanggalSekarang);
  Serial.print("Sensor    : "); Serial.println(SENSOR_LABEL);
  Serial.print("Offset ADC: "); Serial.println(offsetADC, 2);
  Serial.print("RMS ADC   : "); Serial.println(rms, 2);
  Serial.print("VRMS      : "); Serial.println(vrms, 6);
  Serial.print("Noise     : "); Serial.print(dB, 1); Serial.println(" dBA");
  Serial.print("Pesawat   : "); Serial.println(jenisPesawat);
  Serial.print("Status    : "); Serial.println(statusPesawat);
  Serial.print("Kirim ke  : "); Serial.print("http://"); Serial.print(SERVER_IP);
  Serial.print(":"); Serial.print(SERVER_PORT); Serial.println(API_ENDPOINT);
  Serial.println("==========================================");
  Serial.println();
}

//
============================================================
//  SETUP
```

```
//
═══════════════════════════════════════════════════════════════

void setup()
{
  Serial.begin(115200);
  delay(500);

  analogReadResolution(12);
  analogSetAttenuation(ADC_11db);

  Serial.println();

Serial.println("╔═══════════════════════════════════════════════════════════════
");
  Serial.println("║   ANMS – Aircraft Noise Monitoring   ║");
  Serial.println("║   ALAT 1 – Sensor 1                  ║");
  Serial.println("║   Target: MySQL via Node.js API      ║");

Serial.println("╚═══════════════════════════════════════════════════════════════
");
  Serial.println();

  // Koneksi WiFi
  koneksiWiFi();

  // Sinkronisasi waktu NTP
  configTime(GMT_OFFSET_SEC, DAYLIGHT_OFFSET, NTP_SERVER);
  Serial.println("[NTP] Sinkronisasi waktu...");
  delay(2000);
  updateDateTime();
  Serial.print("[NTP] Waktu saat ini: ");
  Serial.print(tanggalSekarang);
  Serial.print(" ");
  Serial.println(jamSekarang);
  Serial.println();
}

//
═══════════════════════════════════════════════════════════════
//  LOOP UTAMA
```

```
//
==========================================================

void loop()
{
  unsigned long now = millis();

  // Kirim data setiap INTERVAL_MS
  if (now - lastSent >= INTERVAL_MS) {
    lastSent = now;

    // Update waktu
    updateDateTime();

    // Hitung noise dari mikrofon
    hitungNoise();

    // Tentukan jenis pesawat dan status
    jenisPesawat  = getPesawat(dB);
    statusPesawat = getStatus(dB);

    // Tampilkan di Serial Monitor
    tampilkanSerial();

    // Kirim ke server MySQL
    kirimKeServer();
  }

  // Reconnect WiFi jika putus
  if (WiFi.status() != WL_CONNECTED) {
    static unsigned long lastReconnect = 0;
    if (now - lastReconnect > 30000) {  // coba reconnect tiap 30 detik
      lastReconnect = now;
      koneksiWiFi();
    }
  }
}
```

Lampiran 3 Program *(Source kode)* Server

```
/**
 * ANMS – Aircraft Noise Monitoring System
 * Express REST API  |  Node.js + MySQL
 *
 * Endpoints
 *
 * POST /api/data          – NodeMCU sends sensor readings
 * GET  /api/data          – Dashboard fetches paginated history
 * GET  /api/data/latest   – Live sensor cards (latest row)
 * GET  /api/data/chart    – Time-series data for graphs
 * GET  /api/stats         – Summary statistics
 * GET  /api/devices       – Registered device list
 * GET  /health            – Health-check / uptime
 */

require('dotenv').config();
const express     = require('express');
const cors        = require('cors');
const rateLimit   = require('express-rate-limit');
const db          = require('./db');

const app  = express();
const PORT = process.env.PORT || 3000;

// ── Middleware ──────────────────────────────────────────
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// CORS
const allowedOrigins = process.env.CORS_ORIGINS
  ? process.env.CORS_ORIGINS.split(',').map(o => o.trim())
  : '*';

app.use(cors({
  origin: allowedOrigins === '*' ? '*' : (origin, cb) => {
    if (!origin || allowedOrigins.includes(origin)) return cb(null, true);
    cb(new Error('Not allowed by CORS'));
```

```
  },
  methods: ['GET', 'POST', 'OPTIONS'],
}));

// Rate limiter for NodeMCU endpoint (generous – 1 req/s per IP)
const deviceLimiter = rateLimit({
  windowMs: 60_000,
  max: 120,
  standardHeaders: true,
  legacyHeaders: false,
  message: { success: false, message: 'Too many requests, slow down.' },
});

// ── Auth middleware for NodeMCU ──────────────────────────────
function requireApiKey(req, res, next) {
  const key = req.headers['x-api-key'] || req.query.api_key;
  if (!key || key !== process.env.DEVICE_API_KEY) {
    return res.status(401).json({ success: false, message: 'Invalid API key' });
  }
  next();
}

// ── Helper ──────────────────────────────────────────────────
function paginate(req) {
  const page  = Math.max(1, parseInt(req.query.page)  || 1);
  const limit = Math.min(100, parseInt(req.query.limit) || 20);
  return { page, limit, offset: (page - 1) * limit };
}

//
// ═══════════════════════════════════════════════════════════
// POST /api/data – NodeMCU sends a reading
//
// ═══════════════════════════════════════════════════════════
app.post('/api/data', deviceLimiter, requireApiKey, async (req, res) => {
  try {
    const {
```

```
    sensor1_db,
    sensor2_db,
    sensor3_db,
    jenis_pesawat = '',
    status        = 'Idle',
    device_id     = 'NodeMCU-01',
  } = req.body;

  // Basic validation
  if ([sensor1_db, sensor2_db, sensor3_db].some(v => v === undefined || isNaN(v))) {
    return res.status(400).json({ success: false, message: 'sensor1_db, sensor2_db, sensor3_db are required numbers' });
  }

  const validStatuses = ['Takeoff', 'Landing', 'Idle'];
  if (!validStatuses.includes(status)) {
    return res.status(400).json({ success: false, message: `status must be one of: ${validStatuses.join(', ')}` });
  }

  const [result] = await db.execute(
    `INSERT INTO sensor_data
      (sensor1_db, sensor2_db, sensor3_db, jenis_pesawat, status, device_id)
     VALUES (?, ?, ?, ?, ?, ?)`,
    [sensor1_db, sensor2_db, sensor3_db, jenis_pesawat, status, device_id]
  );

  // Update last_seen for the device
  await db.execute(
    `UPDATE devices SET last_seen = NOW() WHERE device_id = ?`,
    [device_id]
  );

  res.status(201).json({
    success: true,
    message: 'Data saved',
    id: result.insertId,
  });
} catch (err) {
  console.error('[POST /api/data]', err.message);
  res.status(500).json({ success: false, message: 'Server error' });
```

```
  }
});

//
═══════════════════════════════════════════════════════════════════════
═══════════
//  GET /api/data  –  Paginated history (dashboard table)
//
═══════════════════════════════════════════════════════════════════════
═══════════
app.get('/api/data', async (req, res) => {
try {
    const { page, limit, offset } = paginate(req);
    const { date, device_id, status } = req.query;

    let where = 'WHERE 1=1';
    const params = [];

    if (date) {
      where += ' AND DATE(tanggal_jam) = ?';
      params.push(date);
    }
    if (device_id) {
      where += ' AND device_id = ?';
      params.push(device_id);
    }
    if (status) {
      where += ' AND status = ?';
      params.push(status);
    }

    const [[{ total }]] = await db.execute(
      `SELECT COUNT(*) AS total FROM sensor_data ${where}`, params
    );

    const [rows] = await db.execute(
      `SELECT id, tanggal_jam, sensor1_db, sensor2_db, sensor3_db,
              jenis_pesawat, status, device_id
       FROM sensor_data ${where}
       ORDER BY tanggal_jam DESC
       LIMIT ? OFFSET ?`,
      [...params, limit, offset]
```

```
    );

    res.json({
      success: true,
      data: rows,
      pagination: {
      page,
        limit,
        total,
        totalPages: Math.ceil(total / limit),
      },
    });
  } catch (err) {
    console.error('[GET /api/data]', err.message);
    res.status(500).json({ success: false, message: 'Server error' });
  }
});

//
═══════════════════════════════════════════════════════════════════
//  GET /api/data/latest  –  Most-recent row (live cards)
//
═══════════════════════════════════════════════════════════════════
app.get('/api/data/latest', async (req, res) => {
try {
    const [rows] = await db.execute(
      `SELECT id, tanggal_jam, sensor1_db, sensor2_db, sensor3_db,
            jenis_pesawat, status, device_id
       FROM sensor_data
       ORDER BY tanggal_jam DESC
       LIMIT 1`
    );
    res.json({ success: true, data: rows[0] || null });
  } catch (err) {
    console.error('[GET /api/data/latest]', err.message);
    res.status(500).json({ success: false, message: 'Server error' });
  }
});

//
```

```
// ═══════════════════════════════════════════════════════════════
// GET /api/data/chart – Time-series for graphs
// Query params: hours (default 1), date, hour, minute
// ═══════════════════════════════════════════════════════════════
app.get('/api/data/chart', async (req, res) => {
  try {
    const hours = Math.min(720, parseInt(req.query.hours) || 1);
    const { date, hour, minute } = req.query;

    let baseTime;
    if (date && hour && minute) {
      baseTime = `${date} ${String(hour).padStart(2,'0')}:${String(minute).padStart(2,'0')}:00`;
    } else if (date) {
      baseTime = `${date} 23:59:59`;
    } else {
      baseTime = null; // use NOW()
    }

    const timeExpr = baseTime ? `'${baseTime}'` : 'NOW()';

    const [rows] = await db.execute(
      `SELECT tanggal_jam, sensor1_db, sensor2_db, sensor3_db
       FROM sensor_data
       WHERE tanggal_jam >= DATE_SUB(${timeExpr}, INTERVAL ? HOUR)
         ${baseTime ? `AND tanggal_jam <= '${baseTime}'` : ''}
       ORDER BY tanggal_jam ASC
       LIMIT 1000`,
      [hours]
    );

    res.json({ success: true, data: rows });
  } catch (err) {
    console.error('[GET /api/data/chart]', err.message);
    res.status(500).json({ success: false, message: 'Server error' });
  }
});

//
```

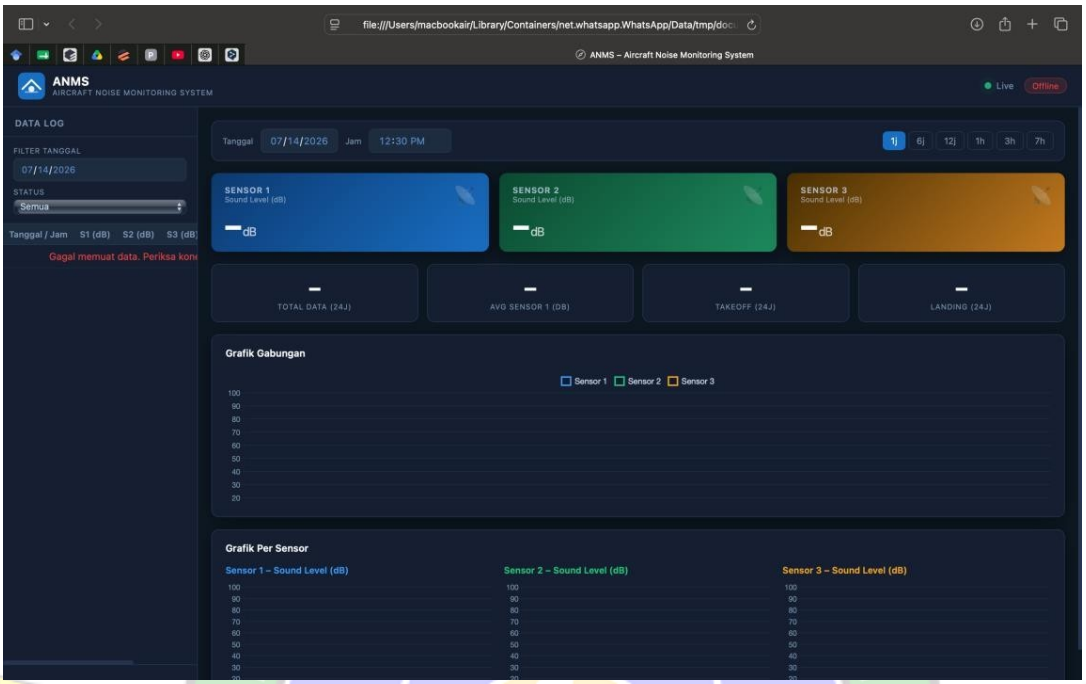
```
// ═══════════════════════════════════════════════════════════════
// GET /api/stats – Summary statistics
// ═══════════════════════════════════════════════════════════════
app.get('/api/stats', async (req, res) => {
try {
  const [[stats]] = await db.execute(`
    SELECT
      COUNT(*)                    AS total_readings,
    ROUND(AVG(sensor1_db), 2)     AS avg_s1,
    ROUND(AVG(sensor2_db), 2)     AS avg_s2,
    ROUND(AVG(sensor3_db), 2)     AS avg_s3,
    ROUND(MAX(sensor1_db), 2)     AS max_s1,
    ROUND(MAX(sensor2_db), 2)     AS max_s2,
    ROUND(MAX(sensor3_db), 2)     AS max_s3,
    ROUND(MIN(sensor1_db), 2)     AS min_s1,
    ROUND(MIN(sensor2_db), 2)     AS min_s2,
    ROUND(MIN(sensor3_db), 2)     AS min_s3,
    SUM(status = 'Takeoff')       AS total_takeoff,
    SUM(status = 'Landing')       AS total_landing
    FROM sensor_data
    WHERE tanggal_jam >= DATE_SUB(NOW(), INTERVAL 24 HOUR)
  `);

  const [byAircraft] = await db.execute(`
    SELECT jenis_pesawat, COUNT(*) AS count
    FROM sensor_data
    WHERE tanggal_jam >= DATE_SUB(NOW(), INTERVAL 24 HOUR)
      AND jenis_pesawat != ''
    GROUP BY jenis_pesawat
    ORDER BY count DESC
  `);

  res.json({ success: true, stats, byAircraft });
 } catch (err) {
  console.error('[GET /api/stats]', err.message);
  res.status(500).json({ success: false, message: 'Server error' });
 }
});
```

```
//
═══════════════════════════════════════════════════════════════════
═══════════════
//  GET /api/devices
//
═══════════════════════════════════════════════════════════════════
═══════════════
app.get('/api/devices', async (req, res) => {
try {
    const [rows] = await db.execute(
      `SELECT device_id, device_name, location, is_active, last_seen, created_at
       FROM devices ORDER BY created_at`
    );
    res.json({ success: true, data: rows });
  } catch (err) {
    console.error('[GET /api/devices]', err.message);
    res.status(500).json({ success: false, message: 'Server error' });
  }
});

//                    ──                    Health                    check
──────────────────────────────────────────
app.get('/health', (_, res) => {
  res.json({ status: 'ok', uptime: process.uptime(), ts: new Date() });
});

//                              ──                    404                    catch-all
──────────────────────────────────────────
app.use((_, res) => res.status(404).json({ success: false, message: 'Route not found' }));

//                                        ──                                        Start
──────────────────────────────────────────
──────
app.listen(PORT, () => {
  console.log(`\n🚀  ANMS Backend running on http://localhost:${PORT}\n`);
});
```

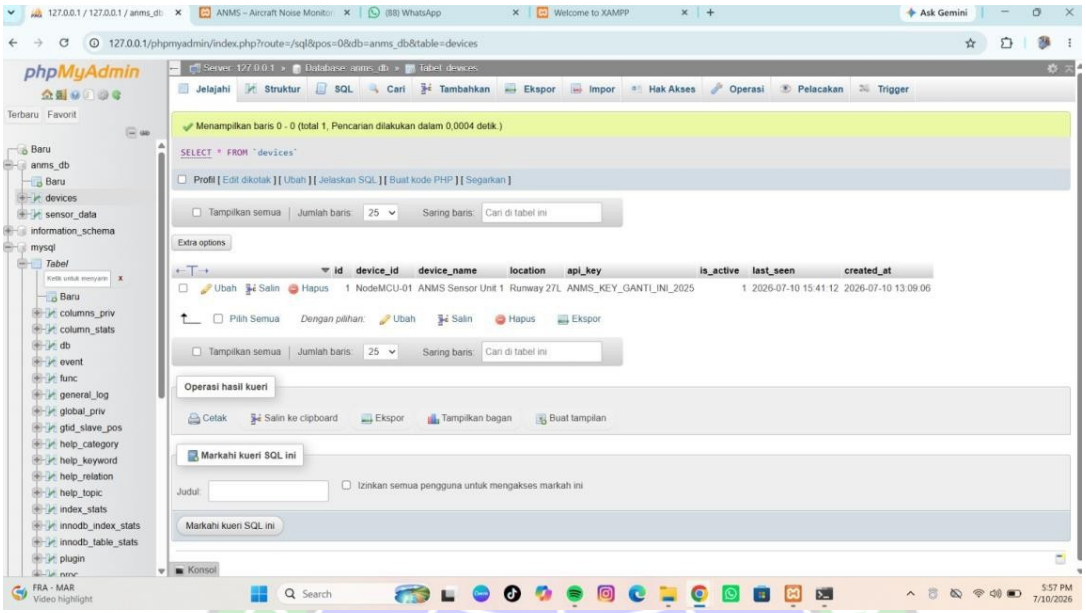
Lampiran 2 Program *(Source kode)* XAMPP

```
/**
 * MySQL connection pool using mysql2/promise
 */
require('dotenv').config();
const mysql = require('mysql2/promise');

const pool = mysql.createPool({
  host              : process.env.DB_HOST     || 'localhost',
  port              : parseInt(process.env.DB_PORT) || 3306,
  user              : process.env.DB_USER     || 'root',
  password          : process.env.DB_PASSWORD || '',
  database          : process.env.DB_NAME     || 'anms_db',
  waitForConnections: true,
  connectionLimit   : 10,
  queueLimit        : 0,
  timezone          : '+00:00',
});

// Test connection on startup
pool.getConnection()
  .then(conn => {
    console.log('■     MySQL connected:', process.env.DB_NAME);
    conn.release();
  })
  .catch(err => {
    console.error('+  MySQL connection failed:', err.message);
    process.exit(1);
  });

module.exports = pool;
```

Lampiran 3 Dashboard Monitoring



Lampiran 4 Database MySQL XAMPP



DAFTAR RIWAYAT HIDUP

Andi Theda Mufarrihah, lahir di Panincong pada tanggal 10 Maret 2005. Anak pertama dari dua bersaudara pasangan Bapak (alm) Anwar dan Ibu Aminah. Bertempat tinggal di Panincong, Desa Panincong, Kecamatan Marioriawa, Kabupaten Soppeng, Provinsi Sulawesi Selatan. Memulai pendidikan di TK Pertiwi Panincong pada tahun 2009 sampai Tahun 2011. Melanjutkan sekolah dasar di SDN 185 Cilellang pada tahun 2011 dan lulus pada tahun 2017. Melanjutkan Sekolah Menengah Pertama di SMPN 1 Soppeng pada Tahun 2017 dan lulus pada Tahun 2020. Melanjutkan sekolah Menengah Atas di SMAN 1 Soppeng pada tahun 2020 dan lulus pada tahun 2023. Selanjutnya pada tahun 2023 diterima sebagai Taruna di Politeknik Penerbangan Surabaya pada Progam Studi Diploma 3 Teknik Pesawat Udara Angkatan 9 sampai saat ini. Selama mengikuti pendidikan di Politeknik Penerbangan Surabaya, penyusun telah mendapatkan kesempatan melaksanakan *On The Job Training* selama satu kali. Penulis melaksanaan *On The Job Training* di PT. Batam *Aero Technic Base Maintenance* dari bulan Januari–Maret 2026.