

APPENDICES

Appendix 1 Expert statement letter

SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Ahmad Daudsyah Imami, S.T., M.T.

Jabatan : Dosen Program Studi Teknik Lingkungan

Instansi : Institut Teknologi Sumatera

No.HP/Email : ahmad.imami@tl.itera.ac.id

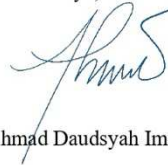
Dengan ini menyatakan bahwa saya benar-benar melakukan proses validasi data terhadap respon AI Gemini milik **I Putu Tude Rama Prasatya**, yang digunakan untuk kepentingan penelitian skripsi dengan judul **“Implementation of MQTT Broker and Gemini API in an Internet of Things (IoT) Based Indoor Air Pollution Monitoring System”**.

Saya melakukan proses validasi tersebut dengan melakukan penilaian dan evaluasi terhadap respons AI Gemini, mencakup aspek kesesuaian, ketepatan, dan relevansi berdasarkan kebutuhan penelitian.

Demikian surat pernyataan ini saya buat dengan sebenar-benarnya untuk digunakan sebagaimana mestinya.

Tempat, Tanggal,

Hormat saya,



Ahmad Daudsyah Imami, S.T., M.T.

Appendix 2 Respondent table

No	Nama	Jenis Kelamin	Email	Jurusan	Universitas	Alamat	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
1	Ni Luh Ade Mulia Dewi	Perempuan	muliadewi280@gmail.com	Pendidikan Ekonomi	Universitas Pendidikan Ganesha	Kost Hijau, Gang Delima 3, Sambangan	4	3	3	4	4	3	3	3	4	4
2	Luh Putu Ranjani Purwa Sanjiwani	Perempuan	ranjanipurwasanjiwani12@gmail.com	Pendidikan Ekonomi	Undiksha	Kost Hijau Delima 3 Sambangan	5	3	5	3	5	5	5	5	5	5
3	Ni Putu Nita Nathalia	Perempuan	nita@student.undiksha.ac.id	S1 Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Gunung Rinjani, Buleleng	5	1	5	1	5	1	5	1	5	1
4	I Komang Wahyu Ambara Putra	Laki - Laki	ambaraputra2004@gmail.com	Ilmu Komputer	Undiksha	Bakti Seraga, Singaraja	5	2	5	2	5	2	5	2	5	2
5	Ananda Adi	Perempuan	anandaadi239@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Asrama Jineng Dalem	4	3	3	2	3	1	4	3	2	2
6	Ni Komang Winduastuti	Perempuan	winduastuti9@gmail.com	Pendidikan Bahasa Inggris	Universitas Pendidikan Ganesha	Kost Delima 3C, Sambangan	5	1	4	3	4	2	3	1	4	4
7	Ivonia Fatima Viegas	Perempuan	ivonia@student.undiksha.ac.id	Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Jatayu no.5 Kaliuntu	4	2	4	1	5	2	4	1	4	1
8	Idalia	Perempuan	idalia@student.Undiksha.ac.id	Ilmu komputer	Universitas Pendidikan Ganesha	Kaliuntu, Edutel	4	4	3	1	4	1	4	1	1	1
9	Achmad Irsyadul Ibad	Laki - Laki	irsyad.ibad2758@gmail.com	Ilmu Komputer	Undiksha	Perum Satelit Asri Banyuasri	3	3	4	2	3	3	2	2	4	3
10	Chiko Gita Satria	Laki - Laki	chikogitasatria23@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Gajah Mada	4	2	4	1	4	2	2	1	4	2
11	I Gusti Nyoman Sapta Wiguna	Laki - Laki	sapta@student.undiksha.ac.id	Pendidikan Teknik Informatika	Universitas Pendidikan Ganesha	Gang Delima, Sambangan	3	2	4	2	4	3	4	2	4	3

No	Nama	Jenis Kelamin	Email	Jurusan	Universitas	Alamat	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
12	Putu Novi Purwaningsih	Perempuan	purwaningsih.novi@student.undiksha.ac.id	Pendidikan Pancasila dan Kewarganegaraan	Universitas Pendidikan Ganesha	Delima 03 Sambangan	2	2	4	5	5	2	4	2	4	4
13	Cindy Hapsari	Perempuan	cindy@student.undiksha.ac.id	Ilmu Komputer	Universitas Pendidikan Ganesha	Banyuning	4	3	4	3	4	3	4	3	4	4
14	Mohamed Izzy Kilian	Laki - Laki	Mohamedizzykilian@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jalan sudirman	4	1	5	1	4	1	4	1	4	4
15	Fieter Brain Pasaribu	Laki - Laki	fieterbrain955@gmail.com	Ilmu Komputer	Undiksha	Baktiseraga	4	2	4	3	5	1	5	2	3	2
16	I Putu Ory Amarika Devananda	Laki - Laki	putuory1@gmail.com	Sistem Informasi	ITB Stikom Bali	Denpasar	4	2	4	3	4	2	3	1	4	3
17	Gede Andra Rizqy Wijaya	Laki - Laki	gedeandrizarizqywijaya@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Kakak Tua No 17	5	2	3	4	5	2	4	2	4	5
18	Ni Putu Lisa Rahmawati	Perempuan	lisarahmawati2010@gmail.com	D3 Perhotelan	Politeknik negeri bali	Jln.pondok Agung	5	5	5	5	5	5	5	5	5	5
19	I Putu Romyadhy Mahaputra	Laki - Laki	romyady221@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jalan Parikesit No. 12, Banjar Tegal, Buleleng, Bali	5	1	5	2	5	4	4	1	5	2
20	Wiswa Nath	Laki - Laki	viswanathsyahputra33@gmail.com	ILKOM	UNDIKSHA	Jln. Wijaya Kusuma no 2-8	4	3	3	1	4	3	4	2	4	2
21	Putu Haryaka Seta Dewa	Laki - Laki	setadewa2004@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Sambangan	4	2	4	1	5	1	4	2	4	3
22	Agung ajus	Laki - Laki	agungajus02@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Jelantik Gingsir No. 2, Sukasada, Buleleng, Bali	4	2	4	1	4	2	3	3	4	5
23	Komang Ratna Mutu Manikam	Perempuan	ratna.mutu@student.undiksha.ac.id	S1 Ilmu Komputer	Universitas Pendidikan Ganesha	Bukti, Kubutambahan, Buleleng.	3	1	5	1	5	1	5	1	5	1
24	Sudita Atmaja	Laki - Laki	sudita098re@gmail.com	S1 Manajemen	Undiksha	Gg.timbul, jl.srikandi, sukasada	4	4	3	4	4	4	3	3	4	4

No	Nama	Jenis Kelamin	Email	Jurusan	Universitas	Alamat	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
25	Krisna	Perempuan	luhmadekrisna@gmail.com	Pendidikan Bahasa Inggris	Universitas Pendidikan Ganesha	Gg. Delima 3, Sambangan	4	3	3	3	4	3	3	3	3	4
26	Komang Pradana Yogi Permana	Laki - Laki	pradanayogi123@gmail.com	Sistem Informasi	Universitas Pendidikan Ganesha	Perumahan Umah Melah Banyuning, Jl. Pulau Kangean, Banyuning	4	4	4	1	4	2	4	2	4	2
27	Kadek Ardy Wirakusuma	Laki - Laki	ardywirakusuma16@gmail.com	Ilmu Komputer	Universitas Pendidikan Ganesha	Jl. Pulau Batam, Banyuning	5	1	5	1	5	1	5	2	5	1
28	Artini Artalia	Perempuan	art.artalia@gmail.com	Sistem Informasi	Universitas Pendidikan Ganesha	Jl. Sudirman, Singaraja	5	2	5	1	4	2	5	1	5	2
29	I Kadek Sindu Arta	Laki - Laki	sindu.arta@student.undiksha	Sistem Informasi	Universitas Pendidikan Ganesha	Jl. Srimadya, banyuning, Buleleng	5	5	5	5	5	5	5	5	5	5
30	Nabila Aulia Azizah	Perempuan	nabila.aulia@student.undiksha.ac.id	Sistem Informasi	Universitas Pendidikan Ganesha	Lovina	4	4	3	2	5	4	4	3	4	4

Appendix 3 Code for the Arduino Nano

```
#include <SoftwareSerial.h>
SoftwareSerial espSerial(10, 11); // RX = D10, TX = D11

// =====
// MQ-7 SENSOR (CO)
// =====

#define MQ7_ANALOG_PIN    A1
#define MQ7_HEATER_PIN    8

float R0 = 35454.0;

// MQ-7 timing
enum MQ7State { HEATING, MEASURING };
MQ7State mq7State = HEATING;

unsigned long stateStart = 0;

// MQ-7 original cycle
const unsigned long HEAT_DURATION    = 60000UL;    // 60 detik
const unsigned long MEASURE_DURATION = 90000UL;    // 90 detik

float lastPPM    = 0;
float lastRatio  = 0;

// =====
// SHARP DUST SENSOR (PM2.5)
// =====

#define SHARP_LED_PIN        2
#define SHARP_ANALOG_PIN    A0

const int samplingTime    = 280;
const int deltaTime       = 40;
const int sleepTime       = 9680;

// ambil sample setiap 15 detik
const unsigned long SHARP_READ_INTERVAL = 15000UL;

// total durasi averaging = 150 detik
const unsigned long SHARP_TOTAL_DURATION = 150000UL;

// hanya mg/m3 yang dirata-rata
float sharpMgSum = 0;
int sharpCount = 0;

unsigned long sharpStartTime = 0;
unsigned long lastSharpRead = 0;

// hasil akhir
float lastDustMg    = 0;
float lastDustUg    = 0;
float lastDustVolt  = 0;

// =====
```

```

// KIRIM DATA KE ESP32
// =====

void sendToESP32() {
  if (lastPPM < 1 && lastDustUg < 1) {
    Serial.println("[SKIP] Data CO dan PM2.5 < 1, tidak
dikirim.");
    return;
  }

  String data =
    "RATIO:" + String(lastRatio, 4) +
    ",PPM:" + String(lastPPM, 2) +
    ",VOLT_PM25:" + String(lastDustVolt, 4) +
    ",MG_PM25:" + String(lastDustMg, 4) +
    ",UG_PM25:" + String(lastDustUg, 2);

  espSerial.println(data);

  Serial.print("[KIRIM] ");
  Serial.println(data);
}

// =====
// SETUP
// =====

void setup() {
  Serial.begin(9600);
  espSerial.begin(9600);

  pinMode(MQ7_HEATER_PIN, OUTPUT);
  pinMode(Sharp_LED_PIN, OUTPUT);

  digitalWrite(MQ7_HEATER_PIN, HIGH);
  digitalWrite(Sharp_LED_PIN, HIGH);

  stateStart = millis();

  sharpStartTime = millis();
  lastSharpRead = millis();

  Serial.println("=====");
  Serial.println(" Sistem Monitoring Udara Dimulai ");
  Serial.println("=====");

  Serial.println("MQ-7: Heating phase...");
}

// =====
// SHARP SENSOR - AMBIL 1 SAMPLE
// =====

void readSharpSample() {
  float voMeasured = 0;
  float calcVoltage = 0;
  float dustDensityMg = 0;

  digitalWrite(Sharp_LED_PIN, LOW);

```

```

delayMicroseconds(samplingTime);

voMeasured = analogRead(SHARP_ANALOG_PIN);

delayMicroseconds(deltaTime);
digitalWrite(SHARP_LED_PIN, HIGH);
delayMicroseconds(sleepTime);

calcVoltage = (voMeasured * 5.0) / 1024.0;

// rumus sensor dust
dustDensityMg = 0.1686 * calcVoltage - 0.0972;

if (dustDensityMg < 0) {
    dustDensityMg = 0;
}

// hanya mg/m3 yang disimpan untuk averaging
sharpMgSum += dustDensityMg;
sharpCount++;

// voltage terakhir hanya untuk ditampilkan
lastDustVolt = calcVoltage;

Serial.print("[Sharp Sample] Volt:");
Serial.print(calcVoltage, 4);
Serial.print(" V | mg/m3: ");
Serial.println(dustDensityMg, 4);
}

// =====
// HITUNG RATA-RATA SHARP
// =====

void calculateSharpAverage() {
    if (sharpCount == 0) return;

    // rata-rata mg/m3
    lastDustMg = sharpMgSum / sharpCount;

    // setelah rata-rata baru dikali 1000 → ug/m3
    lastDustUg = lastDustMg * 1000.0;

    Serial.println("===== RATA-RATA SHARP =====");
    Serial.print("Jumlah Sample : ");
    Serial.println(sharpCount);

    Serial.print("Avg mg/m3      : ");
    Serial.print(lastDustMg, 4);

    Serial.print(" | ug/m3 : ");
    Serial.println(lastDustUg, 2);

    // reset untuk siklus berikutnya
    sharpMgSum = 0;
    sharpCount = 0;
}

// =====

```

```

// MQ-7 SENSOR
// =====

void readMQ7Sensor() {
    int analogValue = analogRead(MQ7_ANALOG_PIN);
    float voltage = analogValue * (5.0 / 1024.0);

    float RL = 10000.0;

    // hitung Rs
    float Rs = (5.0 - voltage) * RL / voltage;

    // ratio
    float ratio = Rs / R0;

    // rumus regresi log
    float a = -236.1;
    float b = 88.522;

    // float ppm = pow(10, a * log10(ratio) + b);

    float ppm = a*ratio+b;

    if (ppm < 0 || isnan(ppm) || isinf(ppm)) {
        ppm = 0;
    }

    lastRatio = ratio;
    lastPPM = ppm;

    Serial.println("===== HASIL MQ-7 =====");
    Serial.print("Analog : ");
    Serial.print(analogValue);

    Serial.print(" | Voltage : ");
    Serial.print(voltage);

    Serial.print(" | Ratio : ");
    Serial.print(ratio, 4);

    Serial.print(" | CO : ");
    Serial.print(ppm, 2);
    Serial.println(" ppm");
}

// =====
// LOOP
// =====

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

    // =====
    // MQ-7 STATE MACHINE
    // =====

    if (mq7State == HEATING) {
        if (now - stateStart >= HEAT_DURATION) {
            digitalWrite(MQ7_HEATER_PIN, LOW);

```

```

        mq7State = MEASURING;
        stateStart = now;

        Serial.println("MQ-7: Heater OFF → Measuring phase...");
    }
}

else if (mq7State == MEASURING) {
    if (now - stateStart >= MEASURE_DURATION) {
        readMQ7Sensor();

        digitalWrite(MQ7_HEATER_PIN, HIGH);

        mq7State = HEATING;
        stateStart = now;

        Serial.println("MQ-7: Heating phase...");
    }
}

// =====
// SHARP SENSOR
// =====

// ambil sample setiap 15 detik
if (now - lastSharpRead >= SHARP_READ_INTERVAL) {
    lastSharpRead = now;
    readSharpSample();
}

// setelah 150 detik → hitung rata-rata + kirim
if (now - sharpStartTime >= SHARP_TOTAL_DURATION) {
    calculateSharpAverage();

    // kirim hasil gabungan ke ESP32
    sendToESP32();

    // reset timer siklus berikutnya
    sharpStartTime = now;
}
}
}

```

Appendix 4 Code for the ESP 32

```

#include <HardwareSerial.h>
#include <WiFi.h>
#include <PubSubClient.h>

// --- WiFi ---
const char* ssid      = "Tude2";
const char* password = "tuderama";

// --- MQTT ---
const char* mqtt_server = "172.20.10.13";
const int  mqtt_port   = 1883;
const char* mqtt_topic  = "sensor/air";

```



```

const char* device_id  = "device_1";

// --- Serial dari Arduino ---
#define RXD2 4
#define TXD2 2

// --- Timing WiFi reconnect ---
unsigned long lastWiFiCheck = 0;
const unsigned long WIFI_CHECK_INTERVAL = 5000UL; // cek tiap 5
detik

WiFiClient  wifiClient;
PubSubClient mqttClient(wifiClient);

// =====

void connectWiFi() {
  if (WiFi.status() == WL_CONNECTED) return; // sudah terhubung,
  skip

  Serial.println("[WiFi] Koneksi terputus, mencoba reconnect...");
  WiFi.disconnect();
  WiFi.begin(ssid, password);

  unsigned long startAttempt = millis();
  Serial.print("[WiFi] Menghubungkan");

  // Tunggu maksimal 10 detik agar loop() tidak terblokir lama
  while (WiFi.status() != WL_CONNECTED && millis() - startAttempt
< 10000UL) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  if (WiFi.status() == WL_CONNECTED) {
    Serial.print("[WiFi] Terhubung! IP: ");
    Serial.println(WiFi.localIP());
  } else {
    Serial.println("[WiFi] Gagal terhubung, akan coba lagi...");
  }
}

void connectMQTT() {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("[MQTT] WiFi tidak terhubung, skip MQTT
reconnect.");
    return;
  }

  int retry = 0;
  while (!mqttClient.connected() && retry < 3) { // max 3x
percobaan
    Serial.print("[MQTT] Menghubungkan...");
    String clientId = "ESP32_" + String(random(0xffff), HEX);
    if (mqttClient.connect(clientId.c_str())) {
      Serial.println("terhubung!");
    } else {
      Serial.print("gagal rc=");

```

```

        Serial.print(mqttClient.state());
        Serial.println(", coba lagi 3 detik...");
        delay(3000);
        retry++;
    }
}

// =====

String getValue(String data, String key, String delim) {
    int startIdx = data.indexOf(key);
    if (startIdx == -1) return "";
    startIdx += key.length();
    int endIdx = data.indexOf(delim, startIdx);
    if (endIdx == -1) return data.substring(startIdx);
    return data.substring(startIdx, endIdx);
}

void parseAndPublish(String data) {
    data.trim();

    if (data.indexOf("RATIO:") == -1 ||
        data.indexOf("PPM:") == -1 ||
        data.indexOf("VOLT_PM25:") == -1 ||
        data.indexOf("MG_PM25:") == -1 ||
        data.indexOf("UG_PM25:") == -1) {
        Serial.println("[ERROR] Format tidak valid: " + data);
        return;
    }

    float ratio = getValue(data, "RATIO:", ",").toFloat();
    float ppm = getValue(data, "PPM:", ",").toFloat();
    float voltPm25 = getValue(data, "VOLT_PM25:", ",").toFloat();
    float mgPm25 = getValue(data, "MG_PM25:", ",").toFloat();
    float ugPm25 = getValue(data, "UG_PM25:", ",").toFloat();

    Serial.println("=====");
    Serial.print("CO Ratio : "); Serial.println(ratio, 4);
    Serial.print("CO PPM : "); Serial.print(ppm, 2);
    Serial.println(" ppm");
    Serial.print("PM2.5 Volt: "); Serial.print(voltPm25, 4);
    Serial.println(" V");
    Serial.print("PM2.5 : "); Serial.print(mgPm25, 4);
    Serial.println(" mg/m3");
    Serial.print("PM2.5 : "); Serial.print(ugPm25, 2);
    Serial.println(" ug/m3");
    Serial.println("=====");

    // Pastikan WiFi & MQTT terhubung sebelum publish
    if (WiFi.status() != WL_CONNECTED) {
        Serial.println("[MQTT] WiFi putus, data tidak dikirim.");
        return;
    }

    if (!mqttClient.connected()) connectMQTT();

    String payload = "{\"device_id\":\"" + String(device_id) + "\",\"
+

```

```

        "\"co\":" + String(ppm, 2) + "," +
        "\"pm_2_5\":" + String(ugPm25, 2) + "}";

    if (mqttClient.publish(mqtt_topic, payload.c_str())) {
        Serial.println("[MQTT] Terkirim: " + payload);
    } else {
        Serial.println("[MQTT] Gagal kirim payload!");
    }
}

// =====

void setup() {
    Serial.begin(115200);
    Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);

    WiFi.mode(WIFI_STA);           // ← pastikan mode station
    WiFi.setAutoReconnect(true);    // ← bawaan ESP32 SDK
    WiFi.persistent(true);          // ← simpan kredensial di flash

    connectWiFi();
    mqttClient.setServer(mqtt_server, mqtt_port);

    Serial.println("=== ESP32 Siap ===");
}

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

    // --- Auto-reconnect WiFi setiap 5 detik jika putus ---
    if (now - lastWiFiCheck >= WIFI_CHECK_INTERVAL) {
        lastWiFiCheck = now;
        if (WiFi.status() != WL_CONNECTED) {
            connectWiFi();
        }
    }

    // --- Auto-reconnect MQTT jika WiFi OK tapi MQTT putus ---
    if (WiFi.status() == WL_CONNECTED && !mqttClient.connected()) {
        connectMQTT();
    }

    mqttClient.loop();

    // --- Baca data dari Arduino ---
    if (Serial2.available()) {
        String incoming = Serial2.readStringUntil('\n');
        if (incoming.length() > 0) {
            parseAndPublish(incoming);
        }
    }
}

```