

LAMPIRAN

Lampiran 1 Surat Kesiediaan Pembimbing 1

SURAT KESEDIAAN MEMBIMBING TA

Yang bertandatangan di bawah ini:

Nama : Rais, S.Pd., M.Kom
NIDN : 0614108501
NIPY : 07.011.083
Jabatan Struktural : Dosen Tetap
Jabatan Fungsional : Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut:

Nama : Rionaldo Jati Garin Putro Raharjo
NIM : 23041041
Program Studi : D-3 Teknik Komputer
Judul TA : RANCANG BANGUN SISTEM ABSENSI
GURU MENGAJAR BERBASIS FINGERPRINT
TERINTEGRASI DENGAN SISTEM
INFORMASI DI SMKN 1 DUKUHTURI

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 28 April 2026

Mengetahui
Ka Prodi DIII Teknik Komputer,



Arif Rakhman, SE, S.Pd, M.Kom.
NIPY. 05.016.291

Dosen Pembimbing I,

Rais, S.Pd., M.Kom
NIDN. 0614108501

Lampiran 2 Surat Kesiediaan Pembimbing 2

SURAT KESEDIAAN MEMBIMBING TA

Yang bertandatangan di bawah ini:

Nama : M. Teguh Prihandoyo, M.Kom.
NIPY : 02.005.012
Jabatan Struktural : Dosen
Jabatan Fungsional : Pengajar

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut:

Nama : Rionaldo Jati Garin Putro Raharjo
NIM : 23041041
Program Studi : D-3 Teknik Komputer
Judul TA : RANCANG BANGUN SISTEM ABSENSI
GURU MENGAJAR BERBASIS FINGERPRINT
TERINTEGRASI DENGAN SISTEM
INFORMASI DI SMKN 1 DUKUHTURI

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 28 April 2026

Mengetahui
Ka Prodi DIII Teknik Komputer,

Dosen Pembimbing II,


Arif Rakhman, S.E., S.Pd., M.Kom.
NIPY. 05.016.291


M. Teguh Prihandoyo, M.Kom.
NIPY. 02.005.012

Lampiran 3 Surat Observasi Tugas Akhir

Nomor : 538.03/KOM-HN/XII/2025
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Kepala SMKN 1 DUKUHTURI

Di Jalan Karanganyar No. 17, Desa Pekauman Kulon, Kecamatan Dukuhturi, Kabupaten Tegal, Jawa Tengah

Dengan Hormat,

Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester V Program Studi Diploma III Teknik Komputer Universitas Harkat Negeri, Maka dengan ini kami mengajukan izin observasi pengambilan data di SMKN 1 DUKUHTURI yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No	NIM	NAMA	NO HP
1	23041038	FIKA AKMAL FUADI	0895379674618
2	23041041	RIONALDO JATI GARIN PUTRO RAHARJO	085727914760

Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 27 April 2026
Ka. Prodi DIII Teknik Komputer
Universitas Harkat Negeri


Universitas
Harkat Negeri
Sekolah Vokasi
Program Studi D-3 Teknik Komputer

Arif Rakhman SE, S.Pd, M.Kom
NIPY. 05.016.291

Lampiran 4 Surat Balasan Observasi Tugas Akhir



PEMERINTAH PROVINSI JAWA TENGAH
DINAS PENDIDIKAN
SEKOLAH MENENGAH KEJURUAN NEGERI 1
DUKUHTURI

Jalan Karanganyar No. 17 ☎(0283) 351533 Fax. 322075 Kabupaten Tegal Kode Pos 52192
Surat Elektronik smkn1dukuhturi@gmail.com



SURAT KETERANGAN
NO. 400.3.8/ 589 /2026

Yang bertanda tangan di bawah ini Kepala SMK Negeri 1 Dukuhturi Kabupaten Tegal, dengan ini menerangkan bahwa :

Nama : FIKA AKMAL FUADI
NIM : 23041038
NO HP : 0895379674618

Nama : RIONALDO JATI GARIN PUTRO RAHARJO
NIM : 23041041
NO HP : 085727914760

telah melaksanakan OBSERVASI PENGAMBILAN DATA untuk kepentingan dalam pembuatan Produk Tugas Akhir di SMK Negeri 1 Dukuhturi Kabupaten Tegal pada tanggal 1 Juni 2026.

Demikian surat keterangan ini dibuat dengan sebenarnya untuk dipergunakan seperlunya.

Dukuhturi, 9 Juni 2026
Kepala SMK Negeri 1 Dukuhturi,



KASLANI, S.Pd., M.Pd.
Pembina
NIP. 197403092010011002

Lampiran 5 Foto Dokumentasi Observasi dan Implementasi

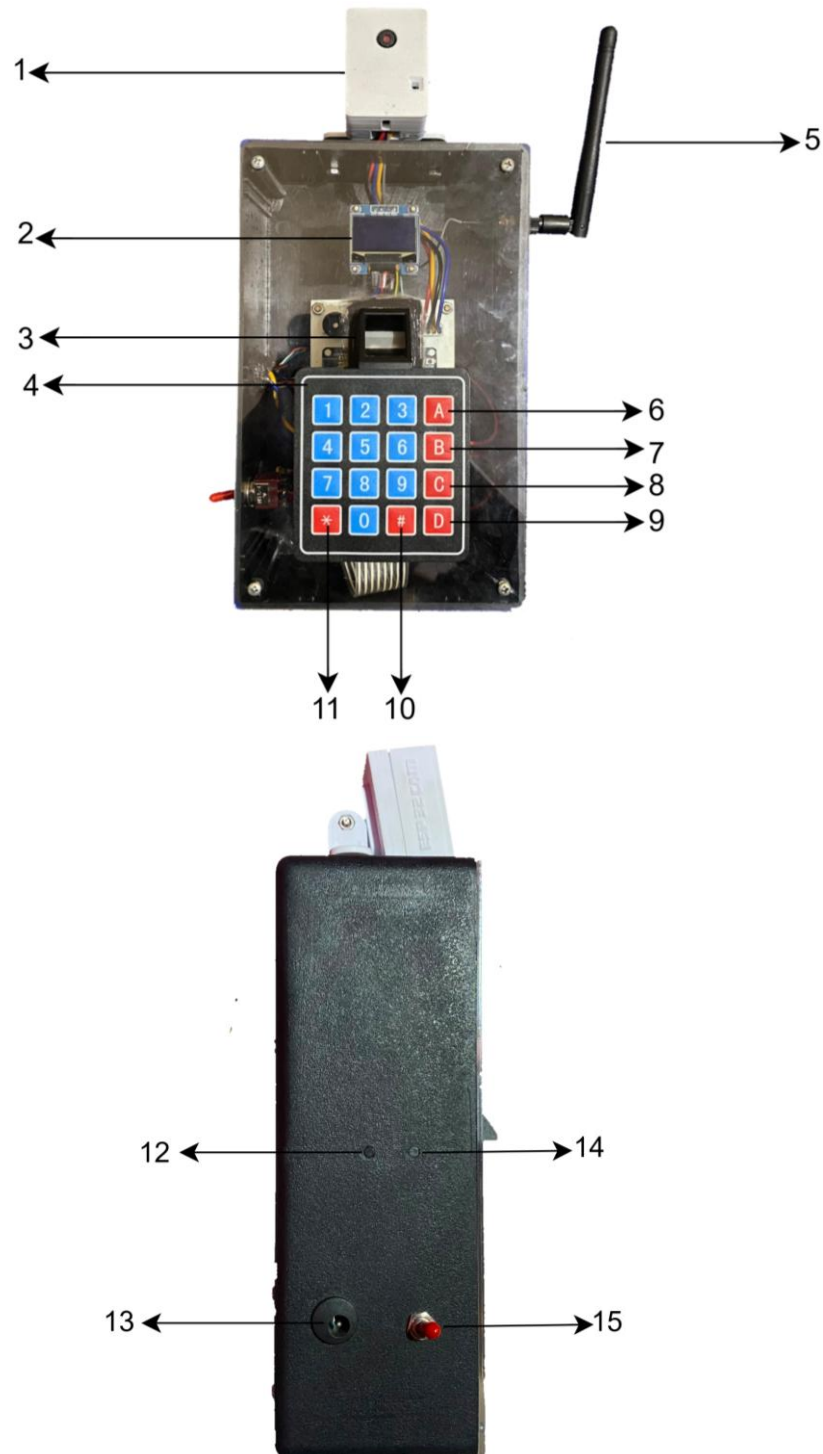
1. Observasi pada Instansi SMKN 1 Dukuhturi



Lampiran 6 Manual Book Penggunaan Alat

MANUAL BOOK PENGGUNAAN ALAT

Sistem Absensi Guru Mengajar Berbasis *Fingerprint* Terintegrasi Dengan
Sistem Informasi di SMKN 1 Dukuhturi



Tabel Keterangan Fungsi Alat

No.	Bagian Alat	Fungsi
1	ESP32-CAM	Digunakan untuk mengambil foto guru saat proses absensi.
2	OLED 0.96	Menampilkan informasi menu, status sistem, data guru, dan proses absensi.
3	Sensor <i>Fingerprint</i> AS608	Digunakan untuk membaca dan mengenali sidik jari guru.
4	Keypad 4x4	Digunakan sebagai media input untuk memilih menu.
5	Antena <i>Wifi</i>	Membantu meningkatkan koneksi perangkat ke jaringan <i>Wifi</i> .
6	Tombol A	Digunakan untuk masuk ke menu absensi.
7	Tombol B	Digunakan untuk masuk ke menu pendaftaran <i>fingerprint</i> (<i>enroll</i>).
8	Tombol C	Digunakan untuk melihat ID <i>fingerprint</i> yang telah terdaftar.
9	Tombol D	Digunakan untuk menghapus ketika salah input ID.
10	Tombol #	Digunakan untuk kembali ke menu atau batal.
11	Tombol *	Digunakan untuk Konfirmasi atau Ya.
12	Tombol Reset Wi-Fi	Digunakan untuk melakukan <i>reset</i> konfigurasi <i>Wifi</i> .
13	Soket Adaptor	Digunakan sebagai konektor sumber tegangan untuk alat.
14	Tombol Delete ID	Digunakan untuk menghapus ID <i>fingerprint</i> yang telah terdaftar.
15	Saklar <i>Switch</i>	Digunakan untuk saklar ON/OFF

Lampiran 7 Code ESP32

```
#include <Wire.h>
#include <Wifi.h>
#include <HTTPClient.h>
#include <WifiClientSecure.h>
#include <time.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_Fingerprint.h>
#include <ArduinoJson.h>
#include <Keypad.h>
#include <WifiManager.h>

#define DEBUG false

String serverURL =
  "https://absensigurusmekar.my.id/api/api_absen.php";
String uploadTemplateURL =
  "https://absensigurusmekar.my.id/api/upload_template.php";
String getTemplatesURL =
  "https://absensigurusmekar.my.id/api/get_templates.php";
String deleteTemplateURL =
  "https://absensigurusmekar.my.id/api/delete_template.php";

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 Display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

HardwareSerial fingerSerial(2);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&fingerSerial);

HardwareSerial camSerial(1);

#define BUZZER 23
#define BTN_DELETE 34
#define BTN_RESETWIFI 39

const byte ROWS = 4;
const byte COLS = 4;

char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

// WIRING KEYPAD
byte rowPins[ROWS] = {32, 33, 25, 26};
byte colPins[COLS] = {27, 14, 12, 13};
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS,
COLS);

bool autoSyncSelesai = false;
bool pendingUpload = false
```

```

int pendingUploadID = 0;
bool sedangScan = false;

// ===== BUZZER =====
void Beep(int d) {
    digitalWrite(BUZZER, HIGH);
    delay(d);
    digitalWrite(BUZZER, LOW);
}

// ===== CENTER TEXT =====
int centerX(String text) {
    return (128 - text.length() * 6) / 2;
}

// ===== OLED HEADER =====
void headerBar() {
    Display.setTextSize(1);
    Display.setTextColor(WHITE);
    Display.setCursor(centerX("ABSENSI GURU MENGAJAR"), 0);
    Display.println("ABSENSI GURU MENGAJAR");
    Display.drawLine(0, 10, 128, 10, WHITE);
}

// ===== OLED TAMPIL =====
void tampil(String a, String b) {
    Display.clearDisplay();
    Display.setTextSize(1);
    Display.setTextColor(WHITE);
    headerBar();
    Display.setCursor(centerX(a), 24);
    Display.println(a);
    Display.setCursor(centerX(b), 40);
    Display.println(b);
    Display.Display();
}

// ===== ERROR FINGERPRINT =====
String errorFingerprint(uint8_t p) {
    if (p == FINGERPRINT_OK) return "OK";
    if (p == FINGERPRINT_PACKETRECEIVEERR) return "PACKET ERROR";
    if (p == FINGERPRINT_NOFINGER) return "NO FINGER";
    if (p == FINGERPRINT_IMAGEFAIL) return "IMAGE FAIL";
    if (p == FINGERPRINT_IMAGEMESS) return "IMAGE MESS";
    if (p == FINGERPRINT_FEATUREFAIL) return "FEATURE FAIL";
    if (p == FINGERPRINT_INVALIDIMAGE) return "INVALID IMG";
    if (p == FINGERPRINT_ENROLLMISMATCH) return "TIDAK COCOK";
    if (p == FINGERPRINT_BADLOCATION) return "ID ERROR";
    if (p == FINGERPRINT_FLASHERR) return "FLASH ERROR";
    if (p == FINGERPRINT_NOTFOUND) return "NOT FOUND";
    return "ERR " + String(p);
}

// ===== HAPUS DIGIT =====
void hapusDigit(String &Input) {

```

```

    if (Input.length() > 0) {
        Input.remove(Input.length() - 1);
        Beep(50);
    }
}

// ===== INPUT ID KEYPAD =====
int InputIDKeypad(String judul) {
    String Input = "";

    while (true) {
        Display.clearDisplay();
        Display.setTextSize(1);
        Display.setTextColor(WHITE);
        headerBar();
        Display.setCursor(centerX(judul), 16);
        Display.println(judul);

        String teks = "ID: " + Input;
        Display.setCursor(centerX(teks), 32);
        Display.println(teks);
        Display.setCursor(centerX("* OK # BACK"), 52);
        Display.println("* OK # BACK");
        Display.Display();
        char key = keypad.getKey();
        if (key) {
            Beep(50);

            if (key >= '0' && key <= '9') {
                if (Input.length() < 3) {
                    Input += key;
                }
            }

            if (key == 'D') {
                hapusDigit(Input);
            }

            if (key == '#') {
                tampil("BATAL", "Kembali");
                delay(1000);
                return -1;
            }
            if (key == '*') {
                if (Input.length() == 0) {
                    tampil("ID KOSONG", "Isi dulu");
                    delay(1000);
                } else {
                    int id = Input.toInt();

                    if (id < 1 || id > 127) {
                        tampil("ID SALAH", "1 - 127");
                        delay(1500);
                        Input = "";
                    }
                }
            }
        }
    }
}

```

```

        } else {
            return id;
        }
    }
}
}
}
delay(50);
}

// ===== JAM MENU =====
void tampilJam() {
    struct tm timeinfo;

    Display.clearDisplay();
    Display.setTextColor(WHITE);

    if (!getLocalTime(&timeinfo)) {
        Display.setTextSize(1);
        Display.setCursor(centerX("ABSENSI GURU MENGAJAR"), 0);
        Display.println("ABSENSI GURU MENGAJAR");

        Display.setCursor(centerX("--/--/----"), 12);
        Display.println("--/--/----");

        Display.drawLine(0, 24, 128, 24, WHITE);

        Display.setTextSize(2);
        Display.setCursor(centerX("--:--"), 36);
        Display.println("--:--");
        Display.Display();
        return;
    }

    char jam[10];
    strftime(jam, sizeof(jam), "%H:%M:%S", &timeinfo);

    char tgl[15];
    strftime(tgl, sizeof(tgl), "%d-%m-%Y", &timeinfo);

    char hari[10];
    strftime(hari, sizeof(hari), "%A", &timeinfo);

    String hariStr = String(hari);

    if (hariStr == "Sunday") hariStr = "Minggu";
    else if (hariStr == "Monday") hariStr = "Senin";
    else if (hariStr == "Tuesday") hariStr = "Selasa";
    else if (hariStr == "Wednesday") hariStr = "Rabu";
    else if (hariStr == "Thursday") hariStr = "Kamis";
    else if (hariStr == "Friday") hariStr = "Jumat";
    else if (hariStr == "Saturday") hariStr = "Sabtu";

    String tanggalLengkap = hariStr + ", " + String(tgl);

```

```

Display.setTextSize(1);
Display.setCursor(centerX("ABSENSI GURU MENGAJAR"), 0);
Display.println("ABSENSI GURU MENGAJAR");
Display.setCursor(centerX(tanggalLengkap), 16);
Display.println(tanggalLengkap);
Display.drawLine(0, 28, 128, 28, WHITE);
Display.setTextSize(2);
Display.setCursor((128 - strlen(jam) * 12) / 2, 40);
Display.println(jam);

    Display.Display();
}

// ===== LOADING SCAN =====
void loadingScan() {
    static int pos = 0;

    Display.clearDisplay();
    Display.setTextColor(WHITE);
    headerBar();
    Display.setTextSize(1);
    Display.setCursor(centerX("Tempel Jari"), 20);
    Display.println("Tempel Jari");
    Display.setCursor(40 + pos, 40);
    Display.print(".");

    pos += 6;
    if (pos > 50) pos = 0;
    Display.Display();
    delay(100);
}

bool tombolResetWifiDitekan() {
    if (digitalRead(BTN_RESETWIFI) == LOW) {
        delay(50); // debounce
        if (digitalRead(BTN_RESETWIFI) == LOW) {
            while (digitalRead(BTN_RESETWIFI) == LOW) {
                delay(10);
            }
            return true;
        }
    }
    return false;
}

// ===== WIFI CONNECT =====
void connectWifi() {
    Wifi.mode(WIFI_STA);
    Wifi.setAutoReconnect(true);
    Wifi.persistent(true);
    Wifi.begin();
    Serial.println("Mencari Wifi tersimpan...");

    unsigned long lastReconnect = 0;

```

```

unsigned long lastDisplay = 0;
int dot = 0;

while (Wifi.status() != WL_CONNECTED) {
    if (tombolResetWifiDitekan()) {
        Serial.println("RESET WIFI MANUAL");
        menuResetWifi();
        return;
    }
    if (millis() - lastReconnect > 10000) {
        Serial.println("Retry Wifi tersimpan...");
        Wifi.disconnect(false, false);
        delay(200);
        Wifi.begin();
        lastReconnect = millis();
    }

    if (millis() - lastDisplay > 1000) {
        String titik = "";
        for (int i = 0; i < dot; i++) titik += ".";
        tampil("Mencari Wifi", String("Reset? Tekan Btn") + titik);
        dot++;
        if (dot > 3) dot = 0;
        lastDisplay = millis();
    }
    delay(100);
    yield();
}
Serial.println("Wifi Terhubung");
Serial.println(Wifi.localIP());
Wifi.setSleep(false);
}

// ===== LIST ID TERPAKAI =====
void listIDTerpakai() {
    tampil("CEK ID", "Mohon Tunggu");
    int ids[127];
    int total = 0;
    while (fingerSerial.available()) {
        fingerSerial.read();
    }
    delay(300);
    for (int i = 1; i <= 127; i++) {
        uint8_t p = finger.loadModel(i);
        if (p == FINGERPRINT_OK) {
            ids[total] = i;
            total++;
        }
        delay(10);
    }

    if (total == 0) {
        tampil("ID TERPAKAI", "Kosong");
        Beep(300);
        delay(2000);
    }
}

```

```

    return;
}

int halaman = 0;
int perHalaman = 12;
int totalHalaman = (total + perHalaman - 1) / perHalaman;

while (true) {
    Display.clearDisplay();
    Display.setTextSize(1);
    Display.setTextColor(WHITE);
    headerBar();

    String judul = "ID TERPAKAI " + String(halaman + 1) + "/" +
String(totalHalaman);
    Display.setCursor(centerX(judul), 13);
    Display.println(judul);

    int start = halaman * perHalaman;
    int end = start + perHalaman;
    if (end > total) end = total;
    int x = 0;
    int y = 25;

    for (int i = start; i < end; i++) {
        Display.setCursor(x, y);
        Display.print(ids[i]);
        x += 32;

        if (x >= 128) {
            x = 0;
            y += 10;
        }
    }

    Display.setCursor(0, 56);
    Display.print("* Next # Back");
    Display.Display();
    char key = keypad.getKey();
    if (key) {
        Beep(50);
        if (key == '#') {
            tampil("KEMBALI", "Menu");
            delay(800);
            return;
        }

        if (key == '*') {
            halaman++;
            if (halaman >= totalHalaman) {
                halaman = 0;
            }
        }
    }
}

```

```

        delay(100);
    }
}

// ===== HAPUS TEMPLATE DI DATABASE =====
bool hapusTemplateDatabase(int fingerID) {
    if (Wifi.status() != WL_CONNECTED) {
        tampil("NO WIFI", "DB Tdk Hapus");
        Beep(500);
        delay(1000);
        return false;
    }

    WifiClientSecure secureClient;
    secureClient.setInsecure();

    HTTPClient http;
    bool beginOK = false;

    if (deleteTemplateURL.startsWith("https://")) {
        beginOK = http.begin(secureClient, deleteTemplateURL);
    } else {
        beginOK = http.begin(deleteTemplateURL);
    }

    if (!beginOK) {
        tampil("DB ERROR", "HTTP Begin");
        Beep(500);
        delay(1000);
        return false;
    }

    http.setTimeout(15000);
    http.setFollowRedirects(HTTPC_STRICT_FOLLOW_REDIRECTS);
    http.addHeader("Content-Type", "application/json");
    http.addHeader("Accept", "application/json");

    String json = "{\"finger_id\":\"" + String(fingerID) + "\"}";

    int httpCode = http.POST(json);
    String response = http.getString();
    http.end();
    if (DEBUG) {
        Serial.print("DELETE DB HTTP: ");
        Serial.println(httpCode);
    }
    if (httpCode != 200) {
        tampil("DB ERROR", "HTTP " + String(httpCode));
        Beep(500);
        delay(1000);
        return false;
    }
    DynamicJsonDocument doc(1024);
    DeserializationError err = deserializeJson(doc, response);

```

```

    if (err) {
        tampil("DB ERROR", "JSON");
        Beep(500);
        delay(1000);
        return false;
    }
    String status = doc["status"] | "";
    return status == "ok";
}

// ===== DELETE FINGERPRINT BY ID =====
void hapusFingerprintID() {
    int id = InputIDKeypad("DELETE FINGER");

    if (id == -1) {
        return;
    }
    tampil("Hapus ID", String(id));
    delay(1000);
    uint8_t p = finger.deleteModel(id);
    if (DEBUG) {
        Serial.print("deleteModel ID ");
        Serial.print(id);
        Serial.print(" = ");
        Serial.println(errorFingerprint(p));
    }

    if (p == FINGERPRINT_OK) {
        tampil("HAPUS SENSOR", "Berhasil");
        Beep(200);
        delay(500);
        tampil("HAPUS DB", "Proses...");
        if (hapusTemplateDatabase(id)) {
            tampil("BERHASIL", "Sensor & DB");
            Beep(200);
            delay(200);
            Beep(200);
        } else {
            tampil("SENSOR OK", "DB Gagal");
            Beep(700);
        }
    } else {
        tampil("GAGAL HAPUS", errorFingerprint(p));
        Beep(700);
    }

    delay(2500);
}

// ===== BASE64 DECODE =====
int base64Value(char c) {
    if (c >= 'A' && c <= 'Z') return c - 'A';
    if (c >= 'a' && c <= 'z') return c - 'a' + 26;
}

```

```

    if (c >= '0' && c <= '9') return c - '0' + 52;
    if (c == '+') return 62;
    if (c == '/') return 63;
    return -1;
}

int base64Decode(String Input, uint8_t *output, int maxLen) {
    int outLen = 0;
    int val = 0;
    int valb = -8;
    Input.trim();
    for (int i = 0; i < Input.length(); i++) {
        char c = Input[i];
        if (c == '=') break;
        if (c == '\r' || c == '\n' || c == ' ') continue;
        int d = base64Value(c);
        if (d < 0) continue;
        val = (val << 6) + d;
        valb += 6;
        if (valb >= 0) {
            if (outLen >= maxLen) return -1;
            output[outLen++] = (uint8_t)((val >> valb) & 0xFF);
            valb -= 8;
        }
    }

    return outLen;
}

// ===== KIRIM COMMAND DOWNCHAR KE AS608
=====
bool kirimCommandDownCharAS608() {
    while (fingerSerial.available()) fingerSerial.read();
    uint8_t cmd[] = {
        0xEF, 0x01,
        0xFF, 0xFF, 0xFF, 0xFF,
        0x01,
        0x00, 0x04,
        0x09, 0x01,
        0x00, 0x0F
    };
    fingerSerial.write(cmd, sizeof(cmd));
    fingerSerial.flush();
    while (fingerSerial.available()) {
        fingerSerial.read();
    }
    uint8_t ack[12];
    int idx = 0;
    unsigned long start = millis();
    while (millis() - start < 2000 && idx < 12) {
        if (fingerSerial.available()) {
            ack[idx++] = fingerSerial.read();
        }
    }
    if (idx < 12) {

```

```

    if (DEBUG) Serial.println("ACK DownChar kurang");
    return false;
}

// ack[9] adalah confirmation code. 0x00 = OK.
if (ack[9] != 0x00) {
    if (DEBUG) {
        Serial.print("DownChar ACK error: ");
        Serial.println(ack[9], HEX);
    }
    return false;
}
return true;
}

// ===== SIMPAN TEMPLATE DATABASE KE SENSOR AS608
=====
// template_data di database berasal dari alat A, berisi packet
template mentah AS608 dalam Base64.
bool simpanTemplateKeSensorAS608(int fingerID, String
templateBase64) {
    if (fingerID < 1 || fingerID > 127) {
        if (DEBUG) Serial.println("fingerID di luar 1-127");
        return false;
    }
    const int MAX_TEMPLATE_PACKET_BYTES = 700;
    uint8_t templatePacket[MAX_TEMPLATE_PACKET_BYTES];
    int templateLen = base64Decode(templateBase64, templatePacket,
MAX_TEMPLATE_PACKET_BYTES);
    if (DEBUG) {
        Serial.print("Decode template ID ");
        Serial.print(fingerID);
        Serial.print(" bytes = ");
        Serial.println(templateLen);
    }
    if (templateLen < 500) {
        return false;
    }
    if (! kirimCommandDownCharAS608()) {
        return false;
    }
    delay(80);
    fingerSerial.write(templatePacket, templateLen);
    fingerSerial.flush();
    delay(300);

    // Bersihkan slot dulu
    finger.deleteModel(fingerID);
    delay(200);

    // Kosongkan serial buffer
    while (fingerSerial.available()) {
        fingerSerial.read();
    }
}

```

```

}
delay(200);
uint8_t p = finger.storeModel(fingerID);
if (DEBUG) {
    Serial.print("storeModel sync ID ");
    Serial.print(fingerID);
    Serial.print(" = ");
    Serial.println(errorFingerprint(p));
}
return p == FINGERPRINT_OK;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

bool uploadTemplateFingerprint(int fingerID) {
    if (Wifi.status() != WL_CONNECTED) {
        tampil("NO WIFI", "Upload Gagal");
        Beep(500);
        delay(1500);
        return false;
    }
    uint8_t p = finger.loadModel(fingerID);
    if (p != FINGERPRINT_OK) {
        tampil("LOAD MODEL", "Gagal");
        return false;
    }
    while (fingerSerial.available()) fingerSerial.read();
    p = finger.getModel();
    if (p != FINGERPRINT_OK) {
        tampil("GET MODEL", "Gagal");
        return false;
    }
}

// ===== BUFFER LEBIH KECIL =====
static uint8_t buffer[512];
int idx = 0;
unsigned long start = millis();
while (millis() - start < 2000 && idx < 512) {
    while (fingerSerial.available()) {
        buffer[idx++] = fingerSerial.read();
        start = millis();
    }
    yield();
    delay(1);
}
if (idx < 100) {
    tampil("TEMPLATE", "Kosong");
    return false;
}
// ===== BASE64 RINGAN =====
String templateBase64;
templateBase64.reserve(700);
const char table[] =
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345678
9+/"

```

```

for (int i = 0; i < idx; i += 3) {
    uint32_t n = ((uint32_t)buffer[i]) << 16;
    if (i + 1 < idx) n |= ((uint32_t)buffer[i + 1]) << 8;
    if (i + 2 < idx) n |= buffer[i + 2];
    templateBase64 += table[(n >> 18) & 63];
    templateBase64 += table[(n >> 12) & 63];
    templateBase64 += (i + 1 < idx) ? table[(n >> 6) & 63] : '=';
    templateBase64 += (i + 2 < idx) ? table[n & 63] : '=';
    yield();
}

WifiClientSecure secureClient;
secureClient.setInsecure();
HTTPClient http;
bool beginOK;
if (uploadTemplateURL.startsWith("https://")) {
    beginOK = http.begin(secureClient, uploadTemplateURL);
} else {
    beginOK = http.begin(uploadTemplateURL);
}
if (!beginOK) {
    tampil("HTTP", "Gagal");
    return false;
}

// ===== JSON MANUAL =====
String json;
json.reserve(900);
json = "{\"finger_id\":\"" + String(fingerID) +
        "\",\"template_data\":\"" + templateBase64 + "\"}";
http.addHeader("Content-Type", "application/json");
yield();
delay(10);
int httpCode = http.POST(json);
yield();
http.end();
if (httpCode == 200) {
    Beep(100);
    delay(100);
    Beep(100);
    templateBase64 = "";
    json = "";
    return true;
}
tampil("UPLOAD", "Gagal");
Beep(500);
delay(1500);
return false;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void hapusIDYangTidakAdaDiDatabase(JsonArray data) {
    bool ada[128];
    for (int i = 0; i < 128; i++) {

```

```

        ada[i] = false;
    }
    // tandai ID yang ADA di database
    for (JsonObject item : data) {
        int id = item["finger_id"] | 0;
        if (id >= 1 && id <= 127) {
            ada[id] = true;
        }
    }

    // cek semua ID lokal sensor
    for (int i = 1; i <= 127; i++) {
        uint8_t p = finger.loadModel(i);
        if (p == FINGERPRINT_OK) {
            // kalau tidak ada di DB → hapus
            if (!ada[i]) {
                tampil("HAPUS ID", String(i));
                finger.deleteModel(i);
                delay(200);
            }
        }
    }
}

// ===== CEK TOTAL DATA DATABASE =====
int getJumlahFingerDatabase() {

    if (Wifi.status() != WL_CONNECTED) {
        return 0;
    }

    WifiClientSecure secureClient;
    secureClient.setInsecure();

    HTTPClient http;

    bool beginOK = false;

    if (getTemplatesURL.startsWith("https://")) {
        beginOK = http.begin(secureClient, getTemplatesURL);
    } else {
        beginOK = http.begin(getTemplatesURL);
    }
    if (!beginOK) {
        return 0;
    }

    http.setTimeout(15000);
    int httpCode = http.GET();
    String response = http.getString();
    http.end();
    if (httpCode != 200) {
        return 0;
    }
    DynamicJsonDocument doc(10000);

```

```

    DeserializationError err = deserializeJson(doc, response);
    if (err) {
        return 0;
    }
    int total = doc["total"] | 0;
    return total;
}

// ===== SYNC SEMUA TEMPLATE DARI DATABASE
=====
bool syncTemplateDariDatabase() {
    if (Wifi.status() != WL_CONNECTED) {
        tampil("NO WIFI", "Sync Gagal");
        Beep(500);
        delay(1500);
        return false;
    }

    WifiClientSecure secureClient;
    secureClient.setInsecure();

    HTTPClient http;
    bool beginOK = false;

    if (getTemplatesURL.startsWith("https://")) {
        beginOK = http.begin(secureClient, getTemplatesURL);
    } else {
        beginOK = http.begin(getTemplatesURL);
    }

    if (!beginOK) {
        tampil("ERROR", "HTTP Begin");
        Beep(500);
        delay(1500);
        return false;
    }

    http.setTimeout(30000);
    http.setFollowRedirects(HTTPC_STRICT_FOLLOW_REDIRECTS);
    http.addHeader("Accept", "application/json");

    int httpCode = http.GET();
    String response = http.getString();
    http.end();

    if (DEBUG) {
        Serial.print("GET template HTTP: ");
        Serial.println(httpCode);
    }

    if (httpCode != 200) {
        Beep(500);
        delay(2000);
        return false;
    }
}

```

```

DynamicJsonDocument doc(12000);
DeserializationError err = deserializeJson(doc, response);

if (err) {
    Beep(500);
    delay(2000);
    return false;
}

String status = doc["status"] | "";
if (status != "ok") {
    tampil("SYNC", "API Error");
    Beep(500);
    delay(2000);
    return false;
}

JsonArray data = doc["data"].as<JsonArray>();
hapusIDYangTidakAdaDiDatabase(data);
int total = data.size();

if (total == 0) {
    tampil("SYNC", "Data Kosong");
    Beep(300);
    delay(2000);
    return true;
}

int sukses = 0;
int gagal = 0;

for (JsonObject item : data) {
    int fingerID = item["finger_id"] | 0;
    String templateData = item["template_data"] | "";

    if (simpanTemplateKeSensorAS608(fingerID, templateData)) {
        sukses++;
        Beep(60);
    } else {
        gagal++;
        Beep(300);
    }

    delay(250);
}

if (gagal == 0) {
} else {
    Beep(500);
}

delay(2500);
return gagal == 0;
}

```

```

// ===== ENROLL DI ALAT B DINONAKTIFKAN
=====
void EnrollFingerprint() {
    if (DEBUG) Serial.println("MODE ENROLL");
    int id = InputIDKeypad("ENROLL ID");

    if (id == -1) return;

    uint8_t cek = finger.loadModel(id);

    if (cek == FINGERPRINT_OK) {
        tampil("ID SUDAH ADA", "Pilih ID Lain");
        Beep(600);
        delay(2000);
        return;
    }

    tampil("ENROLL ID", String(id));
    delay(1000);

    uint8_t p = 0;

    tampil("Tempel Jari", "Pertama");

    while (true) {

        char key = keypad.getKey();

        if (key == '#') {
            tampil("BATAL", "Enroll");
            delay(1000);
            return;
        }

        p = finger.getImage();

        if (p == FINGERPRINT_OK) break;

        delay(50);
    }

    p = finger.image2Tz(1);

    if (p != FINGERPRINT_OK) {
        tampil("Gagal", "Jari Pertama");
        Beep(500);
        delay(2000);
        return;
    }

    tampil("Angkat Jari", "");
    delay(1000);

    while (finger.getImage() != FINGERPRINT_NOFINGER) {

```

```

    delay(50);
}

tampil("Tempel Lagi", "");

while (true) {

    char key = keypad.getKey();

    if (key == '#') {
        tampil("BATAL", "Enroll");
        delay(1000);
        return;
    }

    p = finger.getImage();

    if (p == FINGERPRINT_OK) break;

    delay(50);
}

p = finger.image2Tz(2);

if (p != FINGERPRINT_OK) {
    tampil("Gagal", "Jari Kedua");
    Beep(500);
    delay(1000);
    return;
}

p = finger.createModel();

if (p != FINGERPRINT_OK) {
    tampil("Gagal", "Tidak Cocok");
    Beep(500);
    delay(1000);
    return;
}

p = finger.storeModel(id);
delay(500);

while (fingerSerial.available()) {
    fingerSerial.read();
}

if (p == FINGERPRINT_OK) {

    tampil("Berhasil", "ID: " + String(id));

    Beep(100);
    delay(100);
    Beep(100);

    pendingUploadID = id;
}

```

```

    pendingUpload = true;
    finger.getTemplateCount();
    delay(500);

    while (fingerSerial.available()) {
        fingerSerial.read();
    }
    delay(300);

    return;
} else {

    tampil("Gagal Simpan", errorFingerprint(p));
    Beep(500);
}

    delay(500);
}

// ===== KIRIM KE SERVER ABSENSI =====
bool kirimAbsenKeServer(
    int fingerID,
    String &status,
    String &pesan,
    String &nama,
    String &mapel,
    String &kelas,
    String &jam_mulai,
    String &jam_selesai,
    int &log_id
) {

    if (Wifi.status() != WL_CONNECTED) {
        tampil("NO WIFI", "Absen Gagal");
        Beep(500);
        delay(1500);
        return false;
    }

    WifiClientSecure secureClient;
    secureClient.setInsecure();

    HTTPClient http;

    bool beginOK = false;

    if (serverURL.startsWith("https://")) {
        beginOK = http.begin(secureClient, serverURL);
    } else {
        beginOK = http.begin(serverURL);
    }

    if (!beginOK) {
        tampil("ERROR", "HTTP Begin");
        Beep(500);
    }

```

```

        delay(2000);
        return false;
    }

    http.setTimeout(20000);
    http.setFollowRedirects(HTTPC_STRICT_FOLLOW_REDIRECTS);
    http.addHeader("Content-Type", "application/json");
    http.addHeader("Accept", "application/json");

    String json = "{\"finger_id\":\"" + String(fingerID) + "\"}";

    int httpCode = http.POST(json);
    String response = http.getString();

    http.end();

    if (httpCode <= 0) {
        tampil("ERROR", "Server Down");
        Beep(500);
        delay(2000);
        return false;
    }

    if (httpCode != 200) {
        tampil("ERROR HTTP", String(httpCode));
        Beep(500);
        delay(2000);
        return false;
    }

    DynamicJsonDocument doc(1024);
    DeserializationError error = deserializeJson(doc, response);

    if (error) {
        tampil("ERROR", "JSON Error");
        Beep(500);
        delay(2000);
        return false;
    }

    status = doc["jadwal_status"] | "";
    pesan = doc["pesan"] | "";
    Serial.print("STATUS SERVER: ");
    Serial.println(status);
    nama = doc["guru"] | "";
    mapel = doc["mapel"] | "";
    kelas = doc["kelas"] | "";
    jam_mulai = doc["jam_mulai"] | "";
    jam_selesai = doc["jam_selesai"] | "";
    log_id = doc["log_id"] | 0;

    return true;
}

// ===== TRIGGER ESP32-CAM =====

```

```

void triggerCamera(int fingerID, int log_id) {
    if (log_id <= 0) return;
    while (camSerial.available()) camSerial.read(); // bersihkan
buffer
    String command = "CAPTURE:" + String(fingerID) + "," +
String(log_id);
    camSerial.println(command); // kirim ke ESP32-CAM
    camSerial.flush();
}

// ===== ABSEN =====
void scanAbsensi() {
    if (DEBUG) Serial.println("MODE ABSEN");
    tampil("MODE ABSEN", "Tempel Jari");
    unsigned long startTime = millis();
    while (true) {
        char key = keypad.getKey();

        // ===== CANCEL =====
        if (key == '#') {
            tampil("BATAL", "Absensi");
            delay(500);
            return;
        }

        // ===== TIMEOUT =====
        if (millis() - startTime > 8000) {
            tampil("INFO", "Kembali Menu");
            delay(500);
            return;
        }

        loadingScan();

        // ===== BELUM ADA JARI =====
        if (finger.getImage() != FINGERPRINT_OK) {
            delay(50);
            continue;
        }

        // ===== GAGAL BACA =====
        if (finger.image2Tz() != FINGERPRINT_OK) {
            tampil("GAGAL", "Baca Jari");
            Beep(300);
            delay(1000);
            startTime = millis();
            tampil("MODE ABSEN", "Tempel Jari");
            continue;
        }

        // ===== TIDAK DIKENAL =====
        if (finger.fingerFastSearch() != FINGERPRINT_OK) {
            tampil("GAGAL", "Tidak Dikenal");
            Beep(500);
            delay(1000);
        }
    }
}

```

```

        startTime = millis();
        tampil("MODE ABSEN", "Tempel Jari");
        continue;
    }

    // ===== JARI DITEMUKAN =====
    int id = finger.fingerID;
    tampil("Finger ID", String(id));
    Beep(200);
    delay(500);
    String status = "";
    String nama = "";
    String mapel = "";
    String kelas = "";
    String jam_mulai = "";
    String jam_selesai = "";
    String pesan = "";
    int log_id = 0;

    bool serverOK = kirimAbsenKeServer(
        id,
        status,
        pesan,
        nama,
        mapel,
        kelas,
        jam_mulai,
        jam_selesai,
        log_id
    );

    if (!serverOK) {
        return;
    }
    if (status == "blocked") {
        tampil("INFO", pesan);
        Beep(500);
        delay(2500);
        return;
    }

    // ===== STATUS =====
    if (status == "belum_mulai") {
        tampil("INFO", "Belum Dimulai");
        Beep(500);
        delay(1500);
        return;
    }
    else if (status == "tidak_ada_jadwal") {
        tampil("INFO", "Tidak Ada Jadwal");
        Beep(500);
        delay(1500);
        return;
    }
    else if (status == "sudah_absen") {

```

```

        tampil("ANDA SUDAH", "ABSEN");
        Beep(500);
        delay(1500);
        return;
    }
    else if (status == "finger_tidak_terdaftar") {
        tampil("ERROR", "Finger Tidak Ada");
        Beep(500);
        delay(1500);
        return;
    }
    else if (status == "aktif") {
        // absensi berhasil, tampil data guru seperti sebelumnya
        Display.clearDisplay();
        Display.setTextSize(1);
        Display.setTextColor(WHITE);
        headerBar();
        Display.setCursor(0, 16);
        Display.println(nama.substring(0, 20));
        Display.setCursor(0, 28);
        Display.println(kelas.substring(0, 20));
        Display.setCursor(0, 40);
        Display.println(mapel.substring(0, 20));
        Display.setCursor(0, 52);
        Display.println(jam_mulai + " - " + jam_selesai);
        Display.Display();
        triggerCamera(id, log_id);
        delay(5000);
        Beep(100);
        delay(100);
        return;
    }
    else {
        tampil("STATUS", status);
        Beep(500);
        delay(1500);
    }
}

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

// ===== KONFIRMASI RESET WIFI =====
bool konfirmasiResetWifi() {
    while (true) {
        Display.clearDisplay();
        Display.setTextSize(1);
        Display.setTextColor(WHITE);
        headerBar();
        Display.setCursor(centerX("RESET WIFI?", 20);
        Display.println("RESET WIFI?");
        Display.setCursor(centerX("* YA    # TIDAK", 40);
        Display.println("* YA    # TIDAK");
        Display.Display();
    }
}

```

```

        char key = keypad.getKey();
        if (key) {
            Beep(100);
            if (key == '*') return true;
            if (key == '#') return false;
        }
        delay(50);
    }
}

// ===== MENU RESET WIFI (WRAPPER) =====
void menuResetWifi() {
    if (!konfirmasiResetWifi()) {
        tampil("DIBATALKAN", "");
        delay(800);
        return;
    }
    tampil("RESET WIFI", "Proses...");
    delay(500);
    WifiManager wm;
    wm.ResetSettings();
    wm.startConfigPortal("ABSENSI-SETUP");
    camSerial.println("RESET_WIFI");
    tampil("BERHASIL", "Terhubung");
    delay(1000);
}

void setup() {
    Serial.begin(115200);
    delay(1000);
    if (DEBUG) Serial.println("ESP32 START");
    pinMode(BUZZER, OUTPUT);
    pinMode(BTN_DELETE, INPUT);
    pinMode(BTN_RESETWIFI, INPUT);
    digitalWrite(BUZZER, LOW);

    if (!Display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
        if (DEBUG) Serial.println("OLED gagal!");
        while (1);
    }
    fingerSerial.begin(57600, SERIAL_8N1, 16, 17);
    camSerial.begin(9600, SERIAL_8N1, 18, 19);
    tampil("SYSTEM", "Connecting");
    connectWifi();
    Wifi.setSleep(false);
    configTime(7 * 3600, 0, "pool.ntp.org", "time.google.com");
    delay(1000);

    if (!finger.verifyPassword()) {
        tampil("ERROR", "Fingerprint");
        if (DEBUG) Serial.println("Fingerprint tidak terdeteksi!");
        while (1);
    }

    if (DEBUG) Serial.println("Fingerprint siap!");
}

```

```

    finger.getTemplateCount();
    int totalSensor = finger.templateCount;
    int totalDB = getJumlahFingerDatabase();
    if (DEBUG) {
        Serial.print("Sensor: ");
        Serial.println(totalSensor);

        Serial.print("Database: ");
        Serial.println(totalDB);
    }
    if (totalDB != totalSensor) {
        if (DEBUG) Serial.println("Ada data baru, sync...");
        syncTemplateDariDatabase();
    }
}

// ===== LOOP =====
void loop() {
    char key = keypad.getKey();
    static unsigned long lastWifiReconnect = 0;
    static unsigned long lastWifiDisplay = 0;
    static int WifiDot = 0;

    // ===== WIFI PUTUS =====
    if (Wifi.status() != WL_CONNECTED) {
        if (tombolResetWifiDitekan()) {
            Beep(100);
            sedangScan = true;
            menuResetWifi();
            sedangScan = false;
            return;
        }

        if (millis() - lastWifiReconnect > 10000) {
            Wifi.disconnect(false, false);
            delay(200);
            Wifi.begin();
            lastWifiReconnect = millis();
        }

        if (millis() - lastWifiDisplay > 1000) {
            String titik = "";
            for (int i = 0; i < WifiDot; i++) titik += ".";
            tampil("Mencari Wifi", String("Reset? Tekan Btn") + titik);
            WifiDot++;
            if (WifiDot > 3) WifiDot = 0;
            lastWifiDisplay = millis();
        }
        delay(100);
        return;
    }
    lastWifiReconnect = 0;

    // ===== MENU NORMAL =====

```

```

    if (key && !sedangScan) {
        Beep(50);
        if (key == 'A') {
            sedangScan = true;
            scanAbsensi();
            sedangScan = false;
        }
    }
    else if (key == 'C') {
        sedangScan = true;
        tampil("LIST ID", "Mohon Tunggu");
        delay(500);
        listIDTerpakai();
        sedangScan = false;
    }

    else if (key == '*') {
        tampil("A ABSEN", "C LIST ID");
        delay(700);
    }
}

if (!sedangScan) {
    tampilJam();
}

// ===== UPLOAD BACKGROUND =====
if (pendingUpload && !sedangScan) {
    pendingUpload = false;
    delay(5000);
    uploadTemplateFingerprint(pendingUploadID);
}

// ===== BUTTON ENROLL =====
if (key == 'B' && !sedangScan) { // key adalah hasil keypad.getKey()
    Beep(100);
    delay(300);
    sedangScan = true;
    EnrollFingerprint();
    sedangScan = false;
}

// ===== BUTTON DELETE =====
if (digitalRead(BTN_DELETE) == LOW) {
    delay(50);
    while (digitalRead(BTN_DELETE) == LOW) {
        delay(10);
    }
    Beep(100);
    sedangScan = true;
    tampil("DELETE ID", "Masukkan ID");
    delay(500);
    hapusFingerprintID();
    sedangScan = false;
}

```

```
// ===== BUTTON RESET WIFI =====  
if (tombolResetWifiDitekan()) {  
    Beep(100);  
    sedangScan = true;  
    menuResetWifi();  
    sedangScan = false;  
}  
delay(100);
```