LAMPIRAN

Dokumentasi Penelitian

Sketch Yang digunakan

```
#define BLYNK_PRINT Serial

#define BLYNK_TEMPLATE_ID "TMPL6BYLKnJmY"
#define BLYNK_TEMPLATE_NAME "Smart Trash ESP32"
#define BLYNK_AUTH_TOKEN "Ns2yiNscG6PE8PC9in9zCj3T2nOPqUfE"

#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_TCS34725.h>
#include <ESP32Servo.h>

// ================== WIFI ==================
char ssid[] = "001";
char pass[] = "1spasi23";

// ================== PIN ==================
#define PROX_PIN 27
#define MOISTURE_PIN 34

#define TRIG_ORG 14
#define ECHO_ORG 12
#define TRIG_ANO 26
#define ECHO_ANO 25

#define SERVO_PIN 13

// ===== ULTRASONIC OBJECT =====
#define TRIG_OBJECT 33
```

```
#define ECHO_OBJECT 32

// ================= I2C =================
TwoWire I2C_LCD = TwoWire(1);
LiquidCrystal_I2C lcd(0x27, 16, 2);

// ================= COLOR SENSOR =======
Adafruit_TCS34725 tcs = Adafruit_TCS34725(
  TCS34725_INTEGRATIONTIME_154MS,
  TCS34725_GAIN_16X
);

// ================= SERVO ==============
Servo gateServo;

// ================= BLYNK TIMER =========
BlynkTimer timer;

// ================= VARIABLE ===========
bool metalDetected = false;
bool isWet = false;
bool binFull = false;

String detectedColor = "UNKNOWN";
String trashType = "UNKNOWN";

float refR = 1, refG = 1, refB = 1;

// ================= FUNCTION ===========

void printHeader(String title) {
```

```
  Serial.println("=====================================");
  Serial.println(title);
  Serial.println("=====================================");
}

// ---------- AUTO CALIBRATION ----------
void calibrateColorSensor() {
  Serial.println("⚙️ AUTO CALIBRATION COLOR SENSOR...");
  Serial.println("➡️ Arahkan sensor ke permukaan putih");

  uint32_t sumR = 0, sumG = 0, sumB = 0;
  for (int i = 0; i < 20; i++) {
    uint16_t r, g, b, c;
    tcs.getRawData(&r, &g, &b, &c);
    sumR += r;
    sumG += g;
    sumB += b;
    delay(50);
  }

  refR = sumR / 20.0;
  refG = sumG / 20.0;
  refB = sumB / 20.0;

  Serial.println("✅ Calibration Done\n");
}

// ---------- LCD LOADING BAR ----------
void showLoadingBar() {
  lcd.clear();
```

```
  lcd.setCursor(0, 0);
  lcd.print("Processing...");
  lcd.setCursor(0, 1);
  for (int i = 0; i < 16; i++) {
    lcd.print((char)255);
    delay(90);
  }
}

// ---------- ULTRASONIC ----------
float readUltrasonicCM(int trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH, 30000);
  if (duration == 0) return 999;
  return duration * 0.034 / 2;
}

// ===== ULTRASONIC OBJECT DETECTION =====
float readObjectDistance() {

  digitalWrite(TRIG_OBJECT, LOW);
  delayMicroseconds(2);

  digitalWrite(TRIG_OBJECT, HIGH);
  delayMicroseconds(10);
```

```
  digitalWrite(TRIG_OBJECT, LOW);

  long duration = pulseIn(ECHO_OBJECT, HIGH, 30000);

  if (duration == 0) return 999;

  return duration * 0.034 / 2;
}

bool isBinFull() {
  float distOrg = readUltrasonicCM(TRIG_ORG, ECHO_ORG);
  float distAno = readUltrasonicCM(TRIG_ANO, ECHO_ANO);

  Serial.print("Org Bin Distance: "); Serial.print(distOrg); Serial.println(" cm");
  Serial.print("Ano Bin Distance: "); Serial.print(distAno); Serial.println(" cm");

  return (distOrg < 7 || distAno < 7);
}

// ---------- BIN LEVEL % ----------
int binLevelPercent(float distCM) {
  float emptyDist = 25.0;
  float fullDist  = 7.0;

  distCM = constrain(distCM, fullDist, emptyDist);
  return map(distCM, emptyDist, fullDist, 0, 100);
}

// ---------- COLOR DETECTION ----------
String detectColor(uint16_t r, uint16_t g, uint16_t b, uint16_t c) {
  if (c < 80) return "NO_OBJECT";
```

```
  float nr = r / refR;
  float ng = g / refG;
  float nb = b / refB;

  if (nr > 1.4 && nr > ng && nr > nb) return "RED";
  if (ng > 1.4 && ng > nr && ng > nb) return "GREEN";
  if (nb > 1.4 && nb > nr && nb > ng) return "BLUE";
  if (nr > 1.2 && ng > 1.2 && nb < 1.0) return "YELLOW";
  if (nr > 1.1 && ng > 0.9 && nb < 0.8) return "BROWN";
  if (nr > 0.9 && ng > 0.9 && nb > 0.9) return "WHITE";

  return "OTHER";
}

// ---------- SENSOR READ ----------
void readMetal() {
  metalDetected = digitalRead(PROX_PIN) == LOW;
  Serial.print("Metal Sensor: ");
  Serial.println(metalDetected ? "DETECTED" : "NOT DETECTED");
}

void readMoisture() {
  int moistureValue = analogRead(MOISTURE_PIN);
  isWet = moistureValue < 3500;

  Serial.print("Moisture Value: ");
  Serial.print(moistureValue);
  Serial.print(" -> ");
  Serial.println(isWet ? "WET" : "DRY");
}
```

```
void readColorSensor() {
  uint16_t r, g, b, c;
  tcs.getRawData(&r, &g, &b, &c);
  detectedColor = detectColor(r, g, b, c);

  Serial.print("Color -> ");
  Serial.print("R:"); Serial.print(r);
  Serial.print(" G:"); Serial.print(g);
  Serial.print(" B:"); Serial.print(b);
  Serial.print(" C:"); Serial.print(c);
  Serial.print(" | ");
  Serial.println(detectedColor);
}

// ---------- SERVO CONTROL ----------
void moveServo(String type) {

  if (type == "ORGANIK") {
    Serial.println("Servo → RIGHT (Organik)");
    gateServo.write(150);
  }

  else if (type == "ANORGANIK") {
    Serial.println("Servo → LEFT (Anorganik)");
    gateServo.write(40);
  }

  delay(2000);
  gateServo.write(90);
}
```

```
// ---------- BLYNK SEND ----------
void sendToBlynk() {

  if (!Blynk.connected()) return;

  float distOrg = readUltrasonicCM(TRIG_ORG, ECHO_ORG);
  float distAno = readUltrasonicCM(TRIG_ANO, ECHO_ANO);

  int levelOrg = binLevelPercent(distOrg);
  int levelAno = binLevelPercent(distAno);

  Blynk.virtualWrite(V0, levelOrg);
  Blynk.virtualWrite(V1, levelAno);
  Blynk.virtualWrite(V2, trashType);
  Blynk.virtualWrite(V3, "System Ready");

  static bool orgNotified = false;
  static bool anoNotified = false;

  if (distOrg < 7 && !orgNotified) {
    Blynk.logEvent("organik_penuh", "Tempat sampah ORGANIK penuh!");
    orgNotified = true;
  }

  if (distAno < 7 && !anoNotified) {
    Blynk.logEvent("anorganik_penuh", "Tempat sampah ANORGANIK penuh!");
    anoNotified = true;
  }

  if (distOrg >= 7) orgNotified = false;
```

```
  if (distAno >= 7) anoNotified = false;
}

// ---------- CLASSIFICATION ----------
void classifyTrash() {

  float objectDistance = readObjectDistance();

  if (objectDistance > 10) {

    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Waiting Trash");
    lcd.setCursor(0,1);
    lcd.print("...");

    delay(300);
    return;
  }

  // 1️⃣ Cek bin penuh
  if (isBinFull()) {

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("BIN FULL!");
    lcd.setCursor(0, 1);
    lcd.print("Empty Required");

    Serial.println("🚫 BIN FULL — SYSTEM LOCKED");
```

```
  Serial.println("-------------------------------------");

  if (Blynk.connected()) {
    Blynk.virtualWrite(V3, "BIN FULL");
  }

  delay(1500);
  return;
}

// 2 Cek warna
readColorSensor();

if (detectedColor == "NO_OBJECT") {
  delay(300);
  return;
}

// 3 Loading
showLoadingBar();

// 4 Metal
readMetal();
delay(300);

if (metalDetected) {
  trashType = "ANORGANIK";
  Serial.println(">>> METAL → ANORGANIK");
}
```

```
  else {

    if (detectedColor == "GREEN" || detectedColor == "BROWN" || detectedColor
== "YELLOW" || detectedColor == "RED") {

      readMoisture();

      if (isWet) {
        trashType = "ORGANIK";
        Serial.println(">>> WARNA ORGANIK + LEMBAB → ORGANIK");
      }

      else {
        trashType = "ANORGANIK";
        Serial.println(">>> WARNA ORGANIK TAPI KERING →
ANORGANIK");
      }
    }

    else {
      trashType = "ANORGANIK";
      Serial.println(">>> WARNA NON-ORGANIK → ANORGANIK");
    }
  }

  Serial.print(">>> HASIL AKHIR: ");
  Serial.println(trashType);
  Serial.println("-------------------------------------");

  lcd.clear();
  lcd.setCursor(0,0);
```

```
  lcd.print("Trash Type:");
  lcd.setCursor(0,1);
  lcd.print(trashType);

  moveServo(trashType);

  sendToBlynk();

  delay(1000);
}

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

void setup() {

  Serial.begin(115200);
  delay(500);

  printHeader("SMART TRASH BIN SYSTEM + SERVO + BLYNK");

  pinMode(PROX_PIN, INPUT_PULLUP);

  pinMode(TRIG_OBJECT, OUTPUT);
  pinMode(ECHO_OBJECT, INPUT);

  pinMode(TRIG_ORG, OUTPUT);
  pinMode(ECHO_ORG, INPUT);

  pinMode(TRIG_ANO, OUTPUT);
  pinMode(ECHO_ANO, INPUT);
```

```
Wire.begin(21,22);
I2C_LCD.begin(16,17);

lcd.begin(16,2);
lcd.backlight();

gateServo.attach(SERVO_PIN);
gateServo.write(90);

lcd.setCursor(0,0);
lcd.print("Smart Trash");
lcd.setCursor(0,1);
lcd.print("Calibrating...");
delay(1000);

if (!tcs.begin()) {

  lcd.clear();
  lcd.print("Color ERR");

  Serial.println("❌ TCS34725 not detected!");

  while(1);
}

calibrateColorSensor();

lcd.clear();
lcd.print("Connecting WiFi");

WiFi.begin(ssid, pass);
```

```
unsigned long startAttempt = millis();

while (WiFi.status() != WL_CONNECTED && millis() - startAttempt < 7000) {

  delay(300);
  Serial.print(".");
}

if (WiFi.status() == WL_CONNECTED) {

  Serial.println("\n✅ WiFi Connected");

  Blynk.config(BLYNK_AUTH_TOKEN);
  Blynk.connect(3000);

  if (Blynk.connected()) {
    Serial.println("✅ Blynk Connected");
  }

  else {
    Serial.println("⚠️ Blynk Not Connected (Offline Mode)");
  }
}

else {
  Serial.println("\n⚠️ WiFi Failed (Offline Mode)");
}

timer.setInterval(3000L, sendToBlynk);
```

```
  lcd.clear();
  lcd.print("System Ready");

  delay(1000);
  lcd.clear();
}

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

void loop() {

  if (Blynk.connected()) {
    Blynk.run();
  }

  timer.run();

  classifyTrash();

  delay(300);
}
```