DAFTAR PUSTAKA

Ahmad, F. (2019). Sistem Keamanan Rumah Berbasis Internet Of Things (IoT). Tegal: Politeknik Harapan Bersama.

Anggraini, N., & dkk. (2015). Prototype Alat Monitoring Radioaktivitas Lingkungan, Cuaca dan Kualitas Udara Secara Online dan Periodik Berbasis Arduino (Studi Kasus: Batan Puspiptek Serpong).

Apsari, R. J. (2018). Monitoring Keamanan Rumah Dengan Menggunakan Mikrokontroler Melalui Web. *J. Manaj. Inform*, Vol. 8, No.1, pp 87-95.

Arafat, M. (2016). Sistem Pengaman Pintu Rumah Berbasis Internet Of Things (IoT) Dengan ESP8266. *Jurnal Ilmiah Fakultas Teknik "Technologia"*, Vol. 7, No. 4, pp. 262-268.

Arifianto, D. (2011). *Kamus komponen.* Surabaya: PT Kawan Pustaka.

Baridwan, Z. (2018). *Intermediate Accounting.* Yogyakarta: BPFE-Yogyakarta.

Efendi, M. Y., & Chandra, J. E. (2019). Implementasi Internet of Things pada Sistem Kendali Lampu Rumah Menggunakan Telegram Messenger Bot dan Nodemcu Esp 8266. *Global Journal of Computer Science and Technology*, Vol. 19, No. 1.

Efendi, Y. (2018). Internet Of Things (Iot) Sistem Pengendalian Lampu Menggunakan Raspberry Pi Berbasis Mobile. *Jurnal Ilmiah Ilmu Komputer*, Vol. 4, No.1.

Fani, Handri Al et al. 2020. "Perancangan Alat Monitoring Pendeteksi Suara Di Ruangan Bayi RS Vita Insani Berbasis Arduino Menggunakan Buzzer." *Jurnal Media Informatika Budidarma* 4(1): 144.

F. Dalimarta, F., & dkk. (2022). Rancang Bangun Electronic Control Unit Berbasis Arduino pada Mesin Motor Dua Langkah. *Jurnal Dinamika Vokasional Teknik Mesin*, pp. 105-111.

Karim, R. (2016). Pentingnya Penggunaan Jaringan Wi-Fi Dalam Memenuhi Kebutuhan Informasi Pemustaka pada Kantor Perpustakaan dan Kearsipan Daerah Kotatidore Kepulauan. *Acta Diurna*, pp 2-3.

Mubarok, A., dkk. (2019). Sistem Keamanan Rumah Menggunakan RFID, Sensor PIR dan Modul GSM Berbasis Mikrokontroler. *Jurnal Informatika*, Vol. 5, No.1, pp 137-144.

Muhammad, F. (2016). Pemahaman Tentang "Keamanan Nasional". *Jurnal Ilmu Kepolisian*, pp. 1.

Nasution, M. (2021). Karakteristik Baterai Sebagai Penyimpan Energi Listrik Secara Spesifik. *Journal of Electrical Technology*, Vol. 6, No.1.

Panduardi, F., & Haq, E. (2016). Wireless Smart Home System Menggunakan Raspberry Pi. *Jurnal Teknologi Informasi dan Terapan*, Vol.3, No. 1, pp. 320-325.

Pradipta, R. A. (2023). *Pembuatan Face Recognition System Untuk Absensi Kehadiran Menggunakan Esp32-Cam.* Diploma thesis, Universitas Nasional.

Prastyo, E. A. (2020). *Pengertian, Jenis dan Cara Kerja Kabel Jumper Arduino* . Diakses dari ARDUINO INDONESIA: https://www.arduinoindonesia.id/2022/11/pengertian-jenis-dan-cara-kerja-kabel-jumper arduino.html.

Priyambodo, T. (2005). *Jaringan Wi-Fi, Teori dan Implementasi.* Yogyakarta: Andi.

Salim, P., & Salim, Y. (2002). *Kamus Bahasa Indonesia Kontemporer.* Jakarta: Modern English Press.

Setiawan, A., & Purnamasari, A. I. (2019). Pengembangan Passive Infrared Sensor (PIR) HC-SR501 dengan Microcontrollers ESP32-CAM Berbasiskan Internet of Things (IoT) dan Smart.

Suhardi, D. (2014). Prototipe Controller Lampu Penerangan LED (Light Emitting Diode) Independent Bertenaga Surya. *Jurnal GAMMA*, vol. 10, No.1, pp. 116- 122.

Syaryadhi, M., & dkk. (2007). Sistem Keran Wudhu Menggunakan Sensor PIR Berbasis Mikrokontroler AT89C2051. *Jurnal rekayasa elektrika*, Vol. 6, No. 1.

Tantowi, D., & Kurnia, Y. (2020). Simulasi Sistem Keamanan Kendaraan Roda Dua dengan Smartphone dan GPS Menggunakan Arduino. *Jurnal ARGOR.* pp. 9-15.

Wijaya, E. (2022). *Penggunaan Sensor Ultrasonik pada Sistem Parkir Otomatis Berbasis Mikrokontroler*. Jurnal Elektronika dan Instrumentasi, 9(1), 85-92.

Yunus, M. (2021). *Prototipe Sistem Keamanan Kamar Kos Berbasis Internet of Things menggunakan Sensor Passive Infrared Receiver dengan ESP32-CAM dan Telegram Sebagai Notifikasi (Studi Kasus: Kos Sianturi Air Dingin).* Skripsi. Pekanbaru: Universitas Islam Riau.

LAMPIRAN

Lampiran 1 Program Arduino IDE

```
#include <Arduino.h>
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

const char* ssid = "SMARTHOME";
const char* password = "Smarthome11";

// Initialize Telegram BOT
String BOTtoken = "6566676098:AAF6-
IAzwKJlnJXxLNorC8IyEJKEmdXxVB0";  // your Bot Token (Get from
Botfather)

// Use @myidbot to find out the chat ID of an individual or a group
// Also note that you need to click "start" on a bot before it can
// message you
String CHAT_ID = "7271111550";

bool sendPhoto = false;
int count = 0;
int count1 = 0;
WiFiClientSecure clientTCP;
UniversalTelegramBot bot(BOTtoken, clientTCP);

#define FLASH_LED_PIN 4
bool flashState = LOW;
int pir = 13;
int led = 12;
//Checks for new messages every 1 second.
int botRequestDelay = 1000;
unsigned long lastTimeBotRan;

const int trigPin = 15;
const int echoPin = 14;
const int buzzer = 2;
// Variabel untuk menyimpan waktu terakhir pembacaan
unsigned long previousMillis = 0;
```

```
unsigned long previousMillis1 = 0;

// Interval waktu untuk pembacaan dalam milidetik (misal: setiap
500ms)
const long interval = 100;
const long interval1 = 100;
long duration, distance;

//CAMERA_MODEL_AI_THINKER
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22


void configInitCamera(){
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
```

```
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  //init with high specs to pre-allocate larger buffers
  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;  //0-63 lower number means higher
quality
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;  //0-63 lower number means higher
quality
    config.fb_count = 1;
  }

  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    delay(1000);
    ESP.restart();
  }

  // Drop down frame size for higher initial frame rate
  sensor_t * s = esp_camera_sensor_get();
  s->set_framesize(s, FRAMESIZE_CIF);  //
UXGA|SXGA|XGA|SVGA|VGA|CIF|QVGA|HQVGA|QQVGA
}

void handleNewMessages(int numNewMessages) {
  Serial.print("Handle New Messages: ");
  Serial.println(numNewMessages);

  for (int i = 0; i < numNewMessages; i++) {
    String chat_id = String(bot.messages[i].chat_id);
    if (chat_id != CHAT_ID){
      bot.sendMessage(chat_id, "Unauthorized user", "");
      continue;
    }

    // Print the received message
```

```
    String text = bot.messages[i].text;
    Serial.println(text);
    String welcome = "";
    String from_name = bot.messages[i].from_name;
    if (text == "/start") {
      welcome = "Selamat Datang , " + from_name + "\n";
      welcome += "Gunakan perintah di bawah untuk berinteraksi \n";
      welcome += "/Foto : Mengambil Foto\n";
      welcome += "/flash_Led : Menyalakan LED \n";
      welcome += "/reset : reset sistem \n";
      bot.sendMessage(CHAT_ID, welcome, "");
    }
    if (text == "/flash_Led") {
      flashState = !flashState;
      digitalWrite(FLASH_LED_PIN, flashState);
      Serial.println("Change flash LED state");
      welcome += "Dari ESP32-CAM :\n\n";
      welcome += "LED Flash \n\n";
      welcome += "/start : untuk melihat beberapa perintah.";
      bot.sendMessage(CHAT_ID, welcome, "");
    }
    if (text == "/reset") {
      flashState = !flashState;
      count = 0;
      Serial.println("reset");
      welcome += "Dari ESP32-CAM :\n\n";
      welcome += "Mereset Sistem \n\n";
      welcome += "/start : untuk melihat beberapa perintah.";
      bot.sendMessage(CHAT_ID, welcome, "");
    }
    if (text == "/Foto") {
      sendPhoto = true;
      Serial.println("New photo request");
      welcome += "Dari ESP32-CAM :\n\n";
      welcome += "Mengambil Foto \n\n";
      welcome += "/start : untuk melihat beberapa perintah.";
      bot.sendMessage(CHAT_ID, welcome, "");
    }
  }
}

String sendPhotoTelegram() {
  const char* myDomain = "api.telegram.org";
  String getAll = "";
  String getBody = "";
```

```
  camera_fb_t * fb = NULL;
  fb = esp_camera_fb_get();
  if(!fb) {
    Serial.println("Camera capture failed");
    delay(1000);
    ESP.restart();
    return "Camera capture failed";
  }

  Serial.println("Connect to " + String(myDomain));


  if (clientTCP.connect(myDomain, 443)) {
    Serial.println("Connection successful");

    String head = "--electroniclinic\r\nContent-Disposition: form-
data; name=\"chat_id\"; \r\n\r\n" + CHAT_ID + "\r\n--
electroniclinic\r\nContent-Disposition: form-data; name=\"photo\";
filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
    String tail = "\r\n--electroniclinic--\r\n";

    uint16_t imageLen = fb->len;
    uint16_t extraLen = head.length() + tail.length();
    uint16_t totalLen = imageLen + extraLen;

    clientTCP.println("POST /bot"+BOTtoken+"/sendPhoto HTTP/1.1");
    clientTCP.println("Host: " + String(myDomain));
    clientTCP.println("Content-Length: " + String(totalLen));
    clientTCP.println("Content-Type: multipart/form-data;
boundary=electroniclinic");
    clientTCP.println();
    clientTCP.print(head);

    uint8_t *fbBuf = fb->buf;
    size_t fbLen = fb->len;
    for (size_t n=0;n<fbLen;n=n+1024) {
      if (n+1024<fbLen) {
        clientTCP.write(fbBuf, 1024);
        fbBuf += 1024;
      }
      else if (fbLen%1024>0) {
        size_t remainder = fbLen%1024;
        clientTCP.write(fbBuf, remainder);
      }
```

```
    }

    clientTCP.print(tail);

    esp_camera_fb_return(fb);

    int waitTime = 10000;   // timeout 10 seconds
    long startTimer = millis();
    boolean state = false;

    while ((startTimer + waitTime) > millis()){
      Serial.print(".");
      delay(100);
      while (clientTCP.available()) {
        char c = clientTCP.read();
        if (state==true) getBody += String(c);
        if (c == '\n') {
          if (getAll.length()==0) state=true;
          getAll = "";
        }
        else if (c != '\r')
          getAll += String(c);
        startTimer = millis();
      }
      if (getBody.length()>0) break;
    }
    clientTCP.stop();
    Serial.println(getBody);

  }
  else {
    getBody="Connected to api.telegram.org failed.";
    Serial.println("Connected to api.telegram.org failed.");
  }
  return getBody;

}

void setup(){
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
  // Init Serial Monitor
  Serial.begin(115200);

  // Set LED Flash as output
  pinMode(led, OUTPUT);
```

```
  pinMode(pir, INPUT);
  pinMode(FLASH_LED_PIN, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buzzer, OUTPUT);
  digitalWrite(FLASH_LED_PIN, flashState);

  // Config and init the camera
  configInitCamera();

  // Connect to Wi-Fi
  WiFi.mode(WIFI_STA);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  clientTCP.setCACert(TELEGRAM_CERTIFICATE_ROOT); // Add root
certificate for api.telegram.org
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("ESP32-CAM IP Address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  unsigned long currentMillis = millis();
  unsigned long currentMillis1 = millis();
  int kon = digitalRead(pir);
  Serial.print(kon);Serial.print("\t");Serial.println(count);
  delay(3000);

  if (kon == HIGH){
    sendPhoto = true;
    count++;
    Serial.println("New photo request");
  }

  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    //long duration, distance;
    // Mengirimkan pulsa trigger
    digitalWrite(trigPin, LOW);
```

```
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Membaca pulsa echo
    duration = pulseIn(echoPin, HIGH);

    // Menghitung jarak berdasarkan durasi
    distance = duration * 0.034 / 2;

    // Menampilkan jarak ke serial monitor
    Serial.print("Jarak: ");
    Serial.print(distance);
    Serial.println(" cm");
  }

  if (sendPhoto) {
    if (count > 3){
      digitalWrite(FLASH_LED_PIN, HIGH);
      digitalWrite(led, HIGH);
      Serial.println("Preparing photo");
      sendPhotoTelegram();
      digitalWrite(led, LOW);
      digitalWrite(FLASH_LED_PIN, LOW);
      if(distance <= 50) {
        // Semakin dekat objek, semakin cepat buzzer berbunyi
        int buzzerDelay = map(distance, 7, 50, 255, 70); // Semakin
dekat, semakin pendek delay
        analogWrite(buzzer, buzzerDelay); // Buzzer on
        delay(2000);
        analogWrite(buzzer, 0); // Buzzer off
        delay(1000);
      }//else {digitalWrite(buzzer, LOW);}
      sendPhoto = false;
    }else{
      //digitalWrite(FLASH_LED_PIN, HIGH);
      digitalWrite(led, HIGH);
      Serial.println("Preparing photo");
      sendPhotoTelegram();
      digitalWrite(led, LOW);
      analogWrite(buzzer, 0);
      //digitalWrite(FLASH_LED_PIN, LOW);
      sendPhoto = false;}
  }
```

```
  if (millis() > lastTimeBotRan + botRequestDelay)  {
    int numNewMessages = bot.getUpdates(bot.last_message_received +
1);
    while (numNewMessages) {
      Serial.println("got response");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received +
1);
    }
    lastTimeBotRan = millis();
  }
}
```