KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139. Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

KONSULTASI/ BIMBINGAN TUGAS AKHIR

Nama Mahasiswa : Belinda Aulia Putri
NIM : 062130701712
Jurusan/Program Studi : Teknik Komputer / DIII Teknik Komputer
Judul TA/Skripsi : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android dan Mikrokontroler

NO	TANGGAL	URAIAN	PARAF PEMBIMBING
1.	8 - 3 - 24	Acc Judul	
2.	15 - 3 - 24	Revisi BAB I	
3.	20 - 3 - 24	Revisi BAB I	
4.	24 - 4 - 24	Revisi BAB I	
5.	29 - 4 - 24	Acc BAB I	
6.	8 - 5 - 24	Revisi BAB II	
7.	15 - 5 - 24	Revisi BAB II	
8.	20 - 5 - 24	Acc BAB II	
9.	28 - 5 - 24	Revisi BAB III	
10.	31 - 5 - 24	Acc BAB III & Revisi DP	
11.	3 - 6 - 24	Acc Dapus	
12.	9 - 6 - 24	Acc Proposal.	

Palembang, 2024

Mengetahui,
Ketua Jurusan

Azwardi, S.T., M.T.
NIP. 197005232005011004

KEMENTRIAN PENDIDIKAN DAN KEBUDAYAAN
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TENIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139
Telp. 0711-353414 Fax. 0711-355918
Website: Email:

LEMBAR BIMBINGAN KONSULTASI LAPORAN TUGAS AKHIR

Nama Mahasiswa : Belinda Aulia Putri

NIM : 062130701712

Jurusan/Program Studi : Teknik Komputer/D3 Teknik Komputer

Dosen Pembimbing I : Herlambang Saputra, Ph.D.

Judul Laporan Tugas Akhir : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android Dan Mikrokontroler

NO	TANGGAL	URAIAN	PARAF PEMBIMBING
13	3-7-2024	Revisi Bab IV dan V	
14.	4-7-2024	Acc Bab IV dan V Ujian (a.	

Palembang,
Ketua Jurusan Teknik Komputer

Azwardi, S.T., M.T
NIP. 197705232005501004

KEMENTRIAN PENDIDIKAN DAN KEBUDAYAAN
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TENIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139
Telp. 0711-353414 Fax. 0711-355918
Website: [redacted] Email: [redacted]

LEMBAR BIMBINGAN KONSULTASI LAPORAN TUGAS AKHIR

Nama Mahasiswa : Belinda Aulia Putri

NIM : 062130701712

Jurusan/Program Studi : Teknik Komputer/D3 Teknik Komputer

Dosen Pembimbing II : Arsia Rini, S.Kom., M.Kom.

Judul Laporan Tugas Akhir : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android Dan Mikrokontroler

NO	TANGGAL	URAIAN	PARAF PEMBIMBING
1.	4/6/2024	Bab I	
2.	10/6/2024	Bab II	
3.	14/6/2024	Bab III	
4.	19/6/2024	Bab Hasil Pembahasan (IV)	
5.	27/6/2024	Bab IV dan V	
6.	2/7/2024	Acc ujian LA	

Palembang,
Ketua Jurusan Teknik Komputer

Azwardi, S.T., M.T
NIP. 197705232005010004

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139. Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

REKOMENDASI UJIAN TUGAS AKHIR

Pembimbing Laporan Tugas Akhir, memberikan rekomendasi ujian laporan tugas akhir kepada,

Nama Mahasiswa	:	Belinda Aulia Putri
NIM	:	062130701712
Jurusan/Program Studi	:	Teknik Komputer/D3 Teknik Komputer
Judul Tugas Akhir	:	Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android Dan Mikrokontroler

Mahasiswa tersebut telah memenuhi persyaratan dan dapat mengikuti Ujian Laporan Tugas Akhir, pada Tahun Akademik 2023/2024.

Palembang, Juli 2024

Disetujui oleh,

Pembimbing I	Pembimbing II
Herlambang Saputra, P.hd.	**Arsia Rini, S.Kom., M.Kom.**
NIP. 198103182008121002	**NIP. 198809222020122014**

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139. Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

REVISI UJIAN TUGAS AKHIR

Dosen Penguji : Indarto, S.T., M.Cs.

Nama Mahasiswa : Belinda Aulia Putri

NIM : 062130701712

Jurusan/Program Studi : Teknik Komputer / DIII Teknik Komputer

Judul TA/Skripsi : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android dan Mikrokontroler

No	Uraian	Paraf
		

Palembang,
Dosen Penguji,

(Indarto, S.T., M.Cs.)

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139. Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

REVISI UJIAN TUGAS AKHIR

Dosen Penguji	: Ali Firdaus, M.Kom
Nama Mahasiswa	: Belinda Aulia Putri
NIM	: 062130701712
Jurusan/Program Studi	: Teknik Komputer / DIII Teknik Komputer
Judul TA/Skripsi	: Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android dan Mikrokontroler

No	Uraian	Paraf
	Perbaiki Jurnal Referensi (Jurnal Internasional)	

Palembang,
Dosen Penguji,

(Ali Firdaus, M.Kom)

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139, Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

REVISI UJIAN TUGAS AKHIR

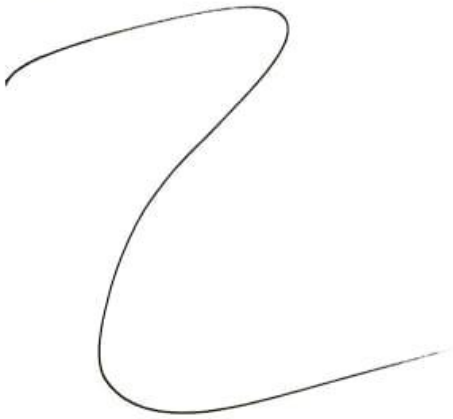
Dosen Penguji : Ica Admirani, M.Kom.

Nama Mahasiswa : Belinda Aulia Putri

NIM : 062130701712

Jurusan/Program Studi : Teknik Komputer / DIII Teknik Komputer

Judul TA/Skripsi : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android dan Mikrokontroler

No	Uraian	Paraf
		

Palembang,
Dosen Penguji,

(Ica Admirani, M.Kom.)

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK KOMPUTER
Jalan Srijaya Negara, Palembang 30139. Telp. 0711-353414
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

PELAKSANAAN REVISI UJIAN TUGAS AKHIR

Nama Mahasiswa : Belinda Aulia Putri
NIM : 062130701712
Jurusan/Program Studi : Teknik Komputer / DIII Teknik Komputer
Judul TA/Skripsi : Perancangan Sistem *Smarthome* Menggunakan *Voice Command* Berbasis Android dan Telegram

Telah melaksanakan revisi terhadap Laporan Tugas Akhir yang diujikan pada hari Senin tanggal 15 bulan Juli tahun 2024 Pelaksanaan revisi terhadap Laporan Tugas Akhir tersebut telah disetujui oleh Dosen Penguji yang memberikan revisi:

No	Komentar	Nama Dosen Penguji	Tanggal/ bulan	Tanda Tangan
1.		Azwardi, S.T., M.T.		
2.	Acc	Ir. A. Bahri Joni M, M.Kom.	23/7/24	
3.	Ok	Indarto S.T., M.Cs.	30/7/24	
4.	Acc	Ali Firdaus, M.Kom	25/7-24	
5.	Acc	Ica Admirani, M.Kom	23/7-2024	

Palembang,
Ketua Penguji

(Azwardi, S.T., M.T.)
NIP : 197005232005011004

LISTING PROGRAM

```
#include <Servo.h>
#include <SoftwareSerial.h>

Servo myServo;  // Membuat objek servo

// Tentukan pin di mana servo terhubung
const int servoPin = D4;  // Pin D4 pada ESP8266

// Tentukan sinyal PWM untuk kontrol servo
const int stopSignal = 1500; // Sinyal PWM untuk menghentikan servo
const int forwardSignal = 1950; // Sinyal PWM untuk memutar maju
const int backwardSignal = 950; // Sinyal PWM untuk memutar mundur

// Bluetooth module pins
const int bluetoothRx = D5; // Bluetooth TX -> ESP8266 RX
const int bluetoothTx = D6; // Bluetooth RX -> ESP8266 TX

SoftwareSerial bluetooth(bluetoothRx, bluetoothTx);

// Lamp control pin
const int lampPin = D1;  // Pin D1 pada ESP8266

void setup() {
  myServo.attach(servoPin);  // Menghubungkan servo ke pin yang telah ditentukan
  myServo.writeMicroseconds(stopSignal); // Pastikan servo dalam posisi berhenti saat
memulai

  bluetooth.begin(9600); // Inisialisasi komunikasi serial Bluetooth
  Serial.begin(115200);  // Inisialisasi komunikasi serial untuk debugging

  pinMode(lampPin, OUTPUT); // Atur pin lampu sebagai output
  digitalWrite(lampPin, LOW); // Pastikan lampu dalam posisi mati saat memulai
}

void loop() {
  if (bluetooth.available()) {
    char command = bluetooth.read(); // Baca perintah dari Bluetooth
    Serial.println(command); // Cetak perintah ke serial monitor untuk debugging
```

```
    switch (command) {
      case 'O': // Perintah buka gerbang
        bukaGerbang();
        Serial.println("gerbang membuka");
        break;
      case 'C': // Perintah tutup gerbang
        tutupGerbang();
        Serial.println("gerbang menutup");
        break;
      case '1': // Perintah hidupkan lampu
        digitalWrite(lampPin, HIGH); // Hidupkan lampu
        Serial.println("lampu hidup");
        break;
      case '0': // Perintah matikan lampu
        digitalWrite(lampPin, LOW); // Matikan lampu
        Serial.println("lampu mati");
        break;
      default:
        Serial.println("Perintah tidak dikenal");
        break;
    }
  }
}

void bukaGerbang() {
  myServo.writeMicroseconds(forwardSignal); // Mengatur servo untuk memutar maju
  delay(3500); // Sesuaikan dengan waktu yang dibutuhkan untuk membuka gerbang 30 cm
  myServo.writeMicroseconds(stopSignal); // Berhenti setelah membuka
}

void tutupGerbang() {
  myServo.writeMicroseconds(backwardSignal); // Mengatur servo untuk memutar mundur
  delay(3600); // Sesuaikan dengan waktu yang dibutuhkan untuk menutup gerbang 30 cm
  myServo.writeMicroseconds(stopSignal); // Berhenti setelah menutup
}
```

```
// Mengimpor perpustakaan yang diperlukan
#include "esp_camera.h"
#include <WiFi.h>
#include "esp_timer.h"
#include "img_converters.h"
#include "Arduino.h"
#include "fb_gfx.h"
#include "soc/soc.h"            // Mengatasi masalah brownout
#include "soc/rtc_cntl_reg.h"    // Mengatasi masalah brownout
#include "esp_http_server.h"
#include <ESP32Servo.h>

// Mengimpor perpustakaan untuk Telegram
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

// Mengganti dengan kredensial jaringan Anda
const char* ssid = "Smarthome";
const char* password = "Smarthome";

// Menginisialisasi bot Telegram
String BOTtoken = "6633998192:AAGAyfpgdlT_tv3Oxt3jT4rZGA2RRlKwLKg";  //
Ganti dengan token bot Anda
String CHAT_ID = "1361404661"; // Ganti dengan chat ID Anda

WiFiClientSecure clientTCP;
UniversalTelegramBot bot(BOTtoken, clientTCP);

#define FLASH_LED_PIN 4
bool flashState = LOW;
bool welcomeMessageSent = false;
bool sendPhoto = false;

// Cek pesan baru setiap 1 detik
int botRequestDelay = 500;
unsigned long lastTimeBotRan;

#define PART_BOUNDARY "123456789000000000000987654321"

#define CAMERA_MODEL_AI_THINKER

// Konfigurasi pin kamera
```

```
#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

#define SERVO_1      14
#define SERVO_STEP   5

Servo servo1;
int servo1Pos = 90; // Posisi awal servo

static const char* _STREAM_CONTENT_TYPE = "multipart/x-mixed-
replace;boundary=" PART_BOUNDARY;
static const char* _STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";
static const char* _STREAM_PART = "Content-Type: image/jpeg\r\nContent-Length:
%u\r\n\r\n";

httpd_handle_t camera_httpd = NULL;
httpd_handle_t stream_httpd = NULL;

// HTML untuk halaman web
static const char PROGMEM INDEX_HTML[] = R"rawliteral(
<!DOCTYPE html>
  <head>
    <title>CCTV SMARTHOME</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style>
     body {
      font-family: 'Arial', sans-serif;
      background-color: #f0f0f0;
```

```
    text-align: center;
    padding-top: 30px;
   }
    img {
    max-width: 100%;
    height: auto;
    display: block;
    margin: 0 auto;
   }
   </style>
  </head>
  <body>
   <h1>CCTV SMARTHOME</h1>
   <img src="/stream" alt="Camera Stream">

<script>
      function sendCommand(direction) {
         var xhr = new XMLHttpRequest();
         xhr.open("GET", "/action?go=" + direction, true);
         xhr.onload = function() {
            if (xhr.readyState == 4 && xhr.status == 200) {
               console.log("Command sent successfully: " + direction);
            } else {
               console.error("Failed to send command: " + direction);
            }
         };
         xhr.onerror = function() {
            console.error("Request failed");
         };
         xhr.send();
      }
   </script>
  </body>
</html>
)rawliteral";

// Handler untuk halaman utama
static esp_err_t index_handler(httpd_req_t *req) {
  httpd_resp_set_type(req, "text/html");
  return httpd_resp_send(req, (const char *)INDEX_HTML, strlen(INDEX_HTML));
}

// Handler untuk streaming kamera
```

```
static esp_err_t stream_handler(httpd_req_t *req) {
   camera_fb_t *fb = NULL;
   esp_err_t res = ESP_OK;
   size_t _jpg_buf_len = 0;
   uint8_t *_jpg_buf = NULL;
    char *part_buf[64];

  res = httpd_resp_set_type(req, "multipart/x-mixed-replace;boundary=frame");
  if (res != ESP_OK) {
    return res;
  }

  while (true) {
    fb = esp_camera_fb_get();
    if (!fb) {
      Serial.println("Gagal mengambil gambar dari kamera");
      res = ESP_FAIL;
        } else {
      if (fb->format == PIXFORMAT_JPEG) {
        _jpg_buf_len = fb->len;
        _jpg_buf = fb->buf;
      } else {
        // convert to jpeg
      }
    }

     if (res == ESP_OK) {
            char part_buf[64];
            size_t hlen = snprintf(part_buf, 64, "--frame\r\nContent-Type:
image/jpeg\r\nContent-Length: %u\r\n\r\n", _jpg_buf_len);
            res = httpd_resp_send_chunk(req, part_buf, hlen);
            res = httpd_resp_send_chunk(req, (const char *)_jpg_buf, _jpg_buf_len);
            res = httpd_resp_send_chunk(req, "\r\n", 2);
            esp_camera_fb_return(fb);
        }

        if (res != ESP_OK) {
            break;
        }
    }

    return res;
}
```

```
// Handler untuk perintah
 esp_err_t cmd_handler(httpd_req_t *req) {
    char* buf;
    size_t buf_len;
    int ret;

    buf_len = httpd_req_get_url_query_len(req) + 1;
    if (buf_len > 1) {
        buf = (char*)malloc(buf_len);
        if (!buf) {
            httpd_resp_send_500(req);
            return ESP_FAIL;
        }
        if (httpd_req_get_url_query_str(req, buf, buf_len) == ESP_OK) {
            if (httpd_query_key_value(buf, "go", buf, buf_len) == ESP_OK) {
                if (strcmp(buf, "left") == 0) {
                    // Logika untuk menggerakkan servo ke kiri
                    if (servo1Pos <= 170) {
                        servo1Pos += SERVO_STEP;
                        servo1.write(servo1Pos);
                        Serial.println("Menggerakkan servo ke kiri");
                        httpd_resp_send(req, "Moved left", HTTPD_RESP_USE_STRLEN);
                    } else {
                        httpd_resp_send(req, "Limit left reached", HTTPD_RESP_USE_STRLEN);
                    }
                } else if (strcmp(buf, "right") == 0) {
                    // Logika untuk menggerakkan servo ke kanan
                    if (servo1Pos >= 10) {
                        servo1Pos -= SERVO_STEP;
                        servo1.write(servo1Pos);
                        Serial.println("Menggerakkan servo ke kanan");
                        httpd_resp_send(req, "Moved right", HTTPD_RESP_USE_STRLEN);
                    } else {
                        httpd_resp_send(req, "Limit right reached",
HTTPD_RESP_USE_STRLEN);
                    }
                } else {
                    httpd_resp_send_404(req);
                }
            } else {
                httpd_resp_send_404(req);
            }
```

```
        } else {
            httpd_resp_send_500(req);
        }
        free(buf);
    } else {
        httpd_resp_send_404(req);
    }
    return ESP_OK;
}



// Memulai server kamera
void startCameraServer() {
  httpd_config_t config = HTTPD_DEFAULT_CONFIG();
  config.server_port = 80;

  httpd_uri_t index_uri = {
    .uri       = "/",
    .method    = HTTP_GET,
    .handler   = index_handler,
    .user_ctx  = NULL
  };

  httpd_uri_t stream_uri = {
    .uri       = "/stream",
    .method    = HTTP_GET,
    .handler   = stream_handler,
    .user_ctx  = NULL
  };

  httpd_uri_t cmd_uri = {
    .uri       = "/action",
    .method    = HTTP_GET,
    .handler   = cmd_handler,
    .user_ctx  = NULL
  };


   if (httpd_start(&camera_httpd, &config) == ESP_OK) {
    Serial.println("Registering index URI");
    httpd_register_uri_handler(camera_httpd, &index_uri);
    Serial.println("Index URI registered");
```

```
    Serial.println("Registering cmd URI");
    httpd_register_uri_handler(camera_httpd, &cmd_uri);
    Serial.println("Cmd URI registered");

    Serial.println("Registering stream URI");
    httpd_register_uri_handler(camera_httpd, &stream_uri);
    Serial.println("Stream URI registered");
  }else {
    Serial.println("Error starting camera server");
  }
}
void sendWelcomeMessage() {
  String welcome = "Selamat datang di CCTV Smarthome! Kamera telah aktif.";
  welcome += "Klik agar perintah sesuai dengan fungsi CCTV:\n";
  welcome += "/photo : perintah mencetak foto dan tersimpan di telegram\n";
  welcome += "/flash : perintah meyalakan dan mematikan Lampu\n";
  welcome += "/right : perintah servo gerak ke kanan \n";
  welcome += "/left : perintah  servo gerak kekiri \n";
  bot.sendMessage(CHAT_ID, welcome, "");
}
// Mengirim pesan ke chat ID tertentu melalui bot Telegram
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;
    }

     String text = bot.messages[i].text;
    Serial.println(text);

    String from_name = bot.messages[i].from_name;
    if (text == "/start") {
      sendWelcomeMessage();
    }
    if (text == "/flash") {
      flashState = !flashState;
      digitalWrite(FLASH_LED_PIN, flashState);
```

```
      bot.sendMessage(chat_id, flashState ? "Flash is ON" : "Flash is OFF", "");
    }
   else if (text == "/photo") {
     sendPhoto = true;
     bot.sendMessage(CHAT_ID, "Sending photo...", "");
    }

    if (text == "/left") {
     if(servo1Pos <= 170) {
     servo1Pos += 10;
     servo1.write(servo1Pos);
     bot.sendMessage(CHAT_ID, "Moving left", "");
     } else {
       bot.sendMessage(CHAT_ID, "Reached left limit", "");
     }
    } else if (text == "/right") {
     if (servo1Pos >= 10) {
       servo1Pos -= 10;
       servo1.write(servo1Pos);
       bot.sendMessage(CHAT_ID, "Moving right", "");
     } else {
       bot.sendMessage(CHAT_ID, "Reached right limit", "");
     }
    }
  }
}
String sendPhotoTelegram() {
  const char* myDomain = "api.telegram.org";
  String getAll = "";
  String getBody = "";

  //Dispose first picture because of bad quality
  camera_fb_t * fb = NULL;
  fb = esp_camera_fb_get();
  esp_camera_fb_return(fb); // dispose the buffered image

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

    size_t imageLen = fb->len;
    size_t extraLen = head.length() + tail.length();
    size_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=RandomNerdTutorials");
    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); // Menonaktifkan
pemadaman brownout

  Serial.begin(115200);
  servo1.attach(SERVO_1, 1000, 2000);
  servo1.write(servo1Pos);

  pinMode(FLASH_LED_PIN, OUTPUT);
  digitalWrite(FLASH_LED_PIN, flashState);

   // Wi-Fi connection
```

```
  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.println("WiFi connected");

  Serial.print("Camera Stream Ready! Go to: http://");
  Serial.println(WiFi.localIP());
  Serial.println("' to connect");
//  Serial.printf("Camera Ready! Use 'http://%s' to connect\n",
WiFi.localIP().toString().c_str());
    if (!welcomeMessageSent) {
     sendWelcomeMessage();
     welcomeMessageSent = true;
     }

  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;

  if (psramFound()) {
    config.frame_size = FRAMESIZE_VGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

  // Menginisialisasi kamera
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  // Start streaming web server
  startCameraServer();

  // Ensure that the clientTCP secure WiFiClient is initialized
  clientTCP.setCACert(TELEGRAM_CERTIFICATE_ROOT);
}
void loop(){
  if (sendPhoto) {
    Serial.println("Preparing photo");
    sendPhotoTelegram();
    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();
    }
    }
```