#define CAMERA\_MODEL\_AI\_THINKER

#include <WiFi.h>

#include <WiFiClientSecure.h>

#include <UniversalTelegramBot.h>

#include "esp\_camera.h"

#include <ArduinoJson.h>

#include "camera\_pins.h"

#include "camera\_code.h"

#include <FS.h>

#include <SPIFFS.h>

#include "SPI.h"

#include "driver/rtc\_io.h"

// Wifi network station credentials

#define WIFI\_SSID "project"

#define WIFI\_PASSWORD "project1234"

// Telegram BOT Token (Get from Botfather)

#define BOT\_TOKEN "5388873550:AAHaCAgnFnIVRqR-QGgXJfaL60YYCCNmcmE"

String CHAT\_ID = "1355785588";

#define FLASH\_LED\_PIN 4

const unsigned long BOT\_MTBS = 100; // mean time between scan messages

unsigned long bot\_lasttime; // last time messages' scan has been done

WiFiClientSecure secured\_client;

UniversalTelegramBot bot(BOT\_TOKEN, secured\_client);

bool flashState = LOW;

camera\_fb\_t \*fb = NULL;

bool isMoreDataAvailable();

byte \*getNextBuffer();

int getNextBufferLen();

bool dataAvailable = false;

String incomingdata;

String rsenvalue;

int sendbit;

#define FILE\_PHOTO "/photo.jpg"

String test\_photo\_url = "https://www.arduino.cc/en/uploads/Trademark/ArduinoCommunityLogo.png";

void handleNewMessages(int numNewMessages)

{

Serial.println("handleNewMessages");

Serial.println(String(numNewMessages));

for (int i = 0; i < numNewMessages; i++)

{

String chat\_id = String(bot.messages[i].chat\_id);

String text = bot.messages[i].text;

Serial.print("---CHAT ID:");Serial.println(chat\_id);Serial.println(text);

String from\_name = bot.messages[i].from\_name;

if (from\_name == "")

from\_name = "Guest";

if (text == "/flash")

{

flashState = !flashState;

digitalWrite(FLASH\_LED\_PIN, flashState);

}

if (text == "/photo")

{

fb = NULL;

// Take Picture with Camera

fb = esp\_camera\_fb\_get();

if (!fb)

{

Serial.println("Camera capture failed");

bot.sendMessage(chat\_id, "Camera capture failed", "");

return;

}

dataAvailable = true;

Serial.println("Sending");

bot.sendPhotoByBinary(chat\_id, "image/jpeg", fb->len,isMoreDataAvailable, nullptr,getNextBuffer, getNextBufferLen);

Serial.println("done!");

esp\_camera\_fb\_return(fb);

}

if (text == "/start")

{

String welcome = "Welcome to the ESP32Cam Telegram bot.\n\n";

welcome += "/photo : will take a photo\n";

welcome += "/flash : toggle flash LED (VERY BRIGHT!)\n";

bot.sendMessage(chat\_id, welcome, "Markdown");

}

}

}

bool isMoreDataAvailable()

{

if (dataAvailable)

{

dataAvailable = false;

return true;

}

else

{

return false;

}

}

byte \*getNextBuffer()

{

if (fb)

{

return fb->buf;

}

else

{

return nullptr;

}

}

int getNextBufferLen()

{

if (fb)

{

return fb->len;

}

else

{

return 0;

}

}

///////////////////////////////// SETUP /////////////////////////////////////

void setup()

{

Serial.begin(9600);

Serial.println();

pinMode(FLASH\_LED\_PIN, OUTPUT);

digitalWrite(FLASH\_LED\_PIN, flashState); //defaults to low

if (!setupCamera())

{

Serial.println("Camera Setup Failed!");

while (true)

{

delay(100);

}

}

// attempt to connect to Wifi network:

Serial.print("Connecting to Wifi SSID ");

Serial.print(WIFI\_SSID);

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

secured\_client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org

while (WiFi.status() != WL\_CONNECTED)

{

Serial.print(".");

delay(500);

}

Serial.print("\nWiFi connected. IP address: ");

Serial.println(WiFi.localIP());

delay(1000);

photsend();

}

void loop()

{

// if (Serial.available() > 0) {

// incomingdata = Serial.readString();

// Serial.print("Read:"); Serial.println(incomingdata);

// incomingdata.trim();

// if (incomingdata.startsWith("#")) {

// Serial.print(" ASH CAME");

// Serial.println(incomingdata);

// incomingdata.remove(0, 1);

// Serial.print("AFTER REMOVE UPDATE INCOMING DATA:"); Serial.println(incomingdata);

// sendbit=incomingdata.toInt();

// if(sendbit==1){

// Serial.println("Preparing photo");

// photsend();

// }

//

// }

//

//}

delay(100);

}

bool checkPhoto( fs::FS &fs ) {

File f\_pic = fs.open( FILE\_PHOTO );

unsigned int pic\_sz = f\_pic.size();

return ( pic\_sz > 100 );

}

// Capture Photo and Save it to SPIFFS

void capturePhotoSaveSpiffs( void ) {

camera\_fb\_t \* fb = NULL; // pointer

bool ok = 0; // Boolean indicating if the picture has been taken correctly

do {

// Take a photo with the camera

Serial.println("Taking a photo...");

fb = esp\_camera\_fb\_get();

if (!fb) {

Serial.println("Camera capture failed");

return;

}

// Photo file name

Serial.printf("Picture file name: %s\n", FILE\_PHOTO);

File file = SPIFFS.open(FILE\_PHOTO, FILE\_WRITE);

// Insert the data in the photo file

if (!file) {

Serial.println("Failed to open file in writing mode");

}

else {

file.write(fb->buf, fb->len); // payload (image), payload length

Serial.print("The picture has been saved in ");

Serial.print(FILE\_PHOTO);

Serial.print(" - Size: ");

Serial.print(file.size());

Serial.println(" bytes");

}

// Close the file

file.close();

esp\_camera\_fb\_return(fb);

// check if file has been correctly saved in SPIFFS

ok = checkPhoto(SPIFFS);

} while ( !ok );

}

void photsend(){

Serial.println("-------------------- TAKE PHOTO AND SEND -----------------");

flashlighton();

delay(1000);

//capturePhotoSaveSpiffs();

delay(1000);

//sendPhotoTelegram();

takephotoandsend();

delay(1000);

flashlightoff();

}

void sendPhotoTelegram(){

String response = bot.sendPhoto(CHAT\_ID, test\_photo\_url, "This photo was sent using URL");

}

void takephotoandsend(){

fb = NULL;

// Take Picture with Camera

fb = esp\_camera\_fb\_get();

if (!fb)

{

Serial.println("Camera capture failed");

bot.sendMessage(CHAT\_ID, "Camera capture failed", "");

return;

}

dataAvailable = true;

Serial.println("Sending");

bot.sendPhotoByBinary(CHAT\_ID, "image/jpeg", fb->len,isMoreDataAvailable, nullptr,getNextBuffer, getNextBufferLen);

Serial.println("done!");

esp\_camera\_fb\_return(fb);

}

void flashlighton(){

digitalWrite(FLASH\_LED\_PIN, HIGH);

}

void flashlightoff(){

digitalWrite(FLASH\_LED\_PIN, LOW);

}