**Arab Open University – Egypt**



**Faculty of Computer Studies**

**Major Networking and Security**

**Building a smart glasses for blind people using artificial intelligence**

**Moheb Youssef Botros Wanes**

**1951510066**

**TM471: Graduation Project, 2022/2023**

**Supervisor: Dr. / Mahmoud Attalla**

**Code of Project: -**

#include "esp\_camera.h"

#include "WiFi.h"

#include "Tesseract.h"

#include "AudioFileSourcePROGMEM.h"

#include "AudioGeneratorWAV.h"

#include "AudioOutputI2S.h"

#include "TTSESP32.h"

// Wi-Fi credentials

const char\* ssid = "ESP32-CAM moheb";

const char\* password = "162000Mhb";

// Pin definitions for the ESP32-CAM module

#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

// Tesseract OCR instance

Tesseract ocr;

// Text to convert to speech

const char\* textToSpeech = "Hello, welcome to the ESP32-CAM text-to-speech example.";

void setup() {

Serial.begin(115200);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

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

delay(1000);

Serial.println("Connecting to WiFi...");

}

// Print the IP address

Serial.println("Connected to WiFi");

Serial.print("IP address: ");

Serial.println(WiFi.localIP());

// Initialize the camera

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\_UXGA;

config.jpeg\_quality = 10;

config.fb\_count = 2;

} else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

// Initialize the camera

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

// Initialize Tesseract OCR

ocr.init("eng"); // Set the language for OCR

Set OCR configuration if needed

ocr.setVariable("key", "value");

Set image resolution for OCR (optional)

ocr.setResolution(300); // Set resolution to 300 DPI

}

// Delay for stability

delay(1000);

}

void loop() {

// Take a picture

camera\_fb\_t\* fb = esp\_camera\_fb\_get();

if (!fb) {

Serial.println("Camera capture failed");

return;

}

// Convert text to speech

TTSESP32 tts;

tts.setLanguage("en");

tts.synthesize(textToSpeech);

// Generate the speech audio

AudioGeneratorWAV \*wav = new AudioGeneratorWAV();

if (wav->begin(&tts, new AudioFileSourcePROGMEM(tts.getBuffer(), tts.getLength()))) {

// Play the speech audio through the speaker

AudioOutputI2S \*out = new AudioOutputI2S();

out->SetPinout(26, 25, 22);

out->SetGain(0.5);

wav->setSampleRate(16000);

out->SetOutputModeMono(true);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

out->SetGain(0.6);

while (wav->isRunning()) {

if (!out->isRunning()) {

wav->stop();

break;

}

delay(10);

}

delete out;

delete wav;

}

//