מגישה: רבקה קליין

ת.ז: 214719650

סמינר בית יעקב רכסים

מנחה: המו' יעל עמר

תאריך הגשה: 15.07.2024 ט' תמוז תשפ"ד

תוכן

מבוא:

רקע לפרויקט:

הפרויקט הוא מכשיר המאפשר אבחון ותרגול של דחיקת לשון שהיא הפרעה בתחום של קלינאית תקשורת.

דחיקת לשון - מצב בו הלשון, שאמורה להיות באחורי הפה, נשענת על השיניים הקדמיות או נכנסת ברווח שבין השיניים העליונות לתחתונות בעת מנוחה, בליעה ולעיתים גם בזמן דיבור.

לכן הסימפטום העיקרי של דחיקת לשון הוא שהלשון נראית מבין השיניים.

הכתובת המתאימה לטיפול עומק בבעיה, היא קלינאית תקשורת שתעבוד עם הילד בצורה סדירה ובאמצעות תרגילים שמטרתם לחזק את שרירי הפנים והפה, ללמוד הגייה נכונה, לתרגל בליעה אפקטיבית של נוזלים ומזון ולאמן את הלשון לשהות במקום הרצוי לנו, באחורי הפה.

התרגול חייב להיות עקבי ומאסיבי. לשם כך חשוב מאוד שהמטופל יהנה מהתרגול ויתמיד בו.

מטרת המערכת:

לאפשר תרגול מהנה וחוויתי. שהמטופל ימשיך ויתמיד בטיפול עד שיגיע לתוצאות טובות, וירגיש צורך ורצון להקדם ולהשתפר.

תיאור הפרויקט:

מכשיר אלקטרוני לאבחון ותרגול עבור הפרעה של דחיקת לשון.

המכשיר מצלם את המטופל כאשר הוא מדבר ומדליק לו נורות בהתאם:

ירוק- אם לא נראתה הלשון.

אדום- אם נראתה מספר רב פעמים.

צהוב- אם לא נראתה. מספר פעמים מועט.

קהל היעד:

כל אחד שסובל מדחיקת לשון, או רוצה לעשות אבחון לא מעמיק של דחיקת לשון.

סקירת ספרות:

בעיה אלגוריתמית:

זיהוי מספר פעמים שהלשון יוצאת:

הסימפטום העיקרי של דחיקת לשון הוא שהלשון מבצבצת בין השיניים. המשתמש מעלה סרטון של המטופל שמדבר וצריך לזהות האם הלשון יוצאת מהשיניים. הזיהוי יעשה ע"י

למידת מכונה עם דאטאסט של תמונות עם לשון בתוך הפה ולשון בחוץ. ויזהה עבור הפריימים של הסרטון האם הלשון בפנים או בחוץ.

כדי להשתמש בזיהוי עצמים של מודל למידת מכונה, צריך מצלמה לקליטת פריימים מהמטופל ושליחת הפריימים שנקלטים לשרת. לאחר מכן לבצע אבחון בשרת עפ"י הזיהוי ולשלוח תשובה למכשיר ולהדליק נורות בהתאם.

השימוש במצלמה מעט מורכב, צריך תנאי תאורה טובים כדי שהמודל יצליח לזהות את הלשון ולהביא תוצאות טובות, בנוסף צריך לוודא שהמטופל עומד במרחק טוב מהמצלמה- לא מידי קרוב ולא מידי רחוק- גם זה עלול להשפיע על תוצאות הזיהוי.

YOLO

YOLO הוא אלגוריתם לזיהוי אובייקטים בזמן אמת שהוצג על ידי Joseph Redmon בשנת 2016.

YOLO מעבד את התמונה כולה במעבר אחד קדימה של רשת עצבית, מה שהופך אותה למהירה במיוחד.

אלגוריתם YOLO מחלק את תמונת הקלט לרשת של תאים ומנבא את ניקוד האובייקט ואת הקואורדינטות
של התיבה התוחמת עבור כל תא.

כל תא אחראי על זיהוי אובייקט מסוים בתמונה.

אלגוריתם YOLO גם מנבא את הסתברויות המחלקה עבור כל אובייקט שזוהה בתמונה.

זה הופך את YOLO לגלאי אובייקטים מרובה מחלקות שיכול לזהות אובייקטים מרובים ממחלקות שונות בתמונה אחת.

YOLO הפכה לבחירה פופולרית עבור יישומים בזמן אמת הדורשים זיהוי אובייקטים מהיר ומדויק.

הפתרון הנבחר:

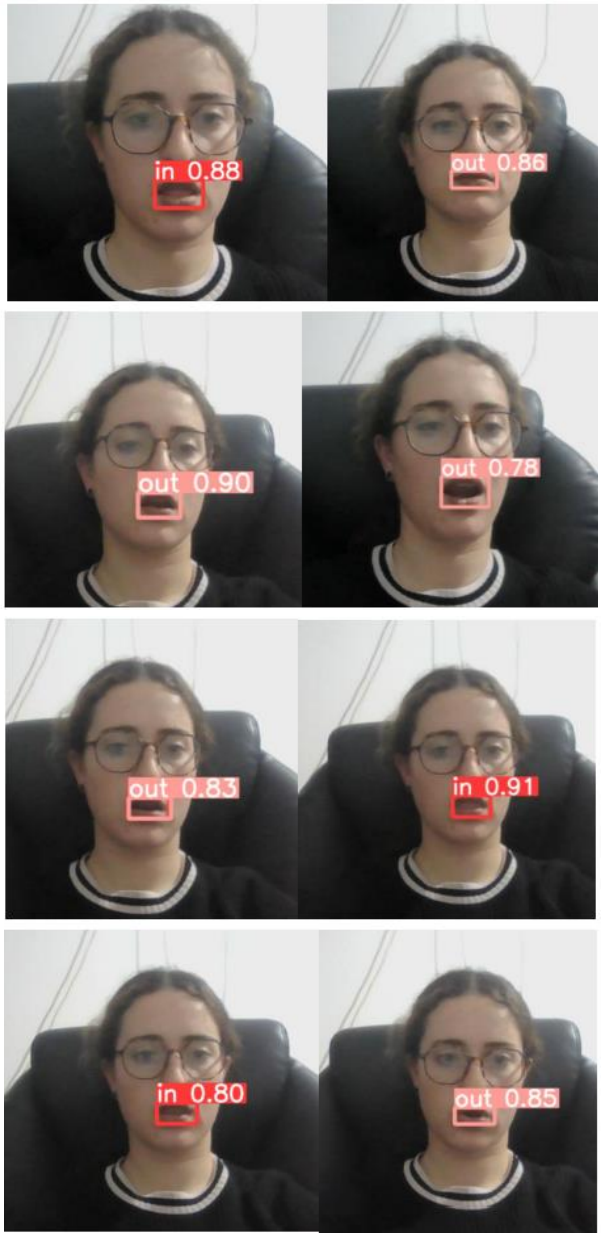
בשלב הראשון המשתמש צריך לפתוח מצלמה, לפני פתיחת המצלמה נבדוק 2 דברים:

1. תאורת החדר באמצעות חיישן LDR.
2. מרחק המטופל מהמצלמה באמצעות חיישן Ultrasonic Sensor.

במקרה שיחזרו נתונים טובים מהחיישנים תפתח מצלמה ESP32-CAM OV2460 והפריימים ישלחו ל #C שממתין לתשובה לבקשות GET ששולח. האבחון יבוצע באמצעות

מודל יולו מאומן על דאטאסט שבניתי של תמונות לשון בפנים ולשון בחוץ. לאחר מכן תשלח תשובה למטופל.

תוצאות זיהוי של היולו:(תגיות מוגדרות הפוך)



פרוט חיישנים ובקרים:

Arduino Uno:

מבנה הלוח:

מיקרו-בקר ATmega328P

המיקרו-בקר המרכזי של הלוח יש לו 32 KB זיכרון פלאש לתכנות, 2 SRAM KB ו- 1 EEPROM KB.

מחבר USB-B

משמש לחיבור הלוח למחשב לצורך תכנות ואספקת חשמל.

כניסות ויציאות דיגיטליות(Digital I/O)

14פינים דיגיטליים ממוספרים מ-0 עד 13. חלקם תומכים ב PWM פלט אות מרוחב משתנה.

כניסות אנלוגיות(Analog Inputs)

6פינים אנלוגיים ממוספרים מ- A0 עד A5. מאפשרים לקרוא אותות אנלוגיים ולבצע המרה דיגיטלית.

מחבר חשמל(Power Jack)

מחבר לכניסת מתח חיצוני של 7-12 V.

פין V5 ופין V3.3

פינים המספקים מתח קבוע של V5 ו-V3.3 למתן כוח לרכיבים חיצוניים.

מחבר ICSP

מחבר המשמש לתכנות המיקרו-בקר על הלוח באמצעות כבל ICSP In-Circuit Serial Programming.

כפתור Reset

כפתור לאיפוס המיקרו-בקר והפעלת הקוד מחדש.

מאפיינים טכניים:

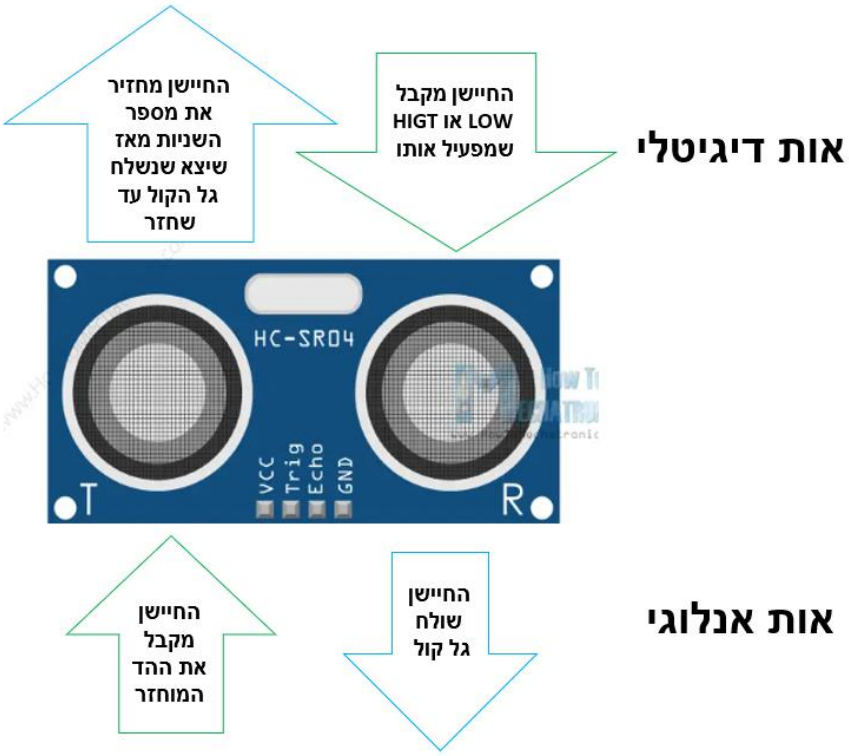
- **מתח הפעלה:** 5V
- **מתח כניסה מומלץ:** 7-12V
- **כניסות ויציאות דיגיטליות:** 14
- **כניסות אנלוגיות:**6
- **זרם מקסימלי ל- Pin I/O:** 40mA
- **זיכרון פלאש:** 32KB
- **SRAM**: 2KB
- **EEPROM**: 1KB
- **מהירות שעון:** 16MHz

חיישן מרחק:

חיישן המרחק האולטראסוני הוא מכשיר המשתמש בגלי קול בתדרים גבוהים מהטווח הנשמע של בני אדם (בד"כ מעל 20 קילו-הרץ) כדי לזהות ולמדוד מרחקים לעצמים. הוא פועל על בסיס עיקרון ההד, בדומה לאופן שבו העטלפים והדולפינים מנווטים ומזהים עצמים בסביבתם.

איך עובד:

1. המשדר פולט פרץ של גלים קוליים.
2. גלים אלו נעים באוויר ופוגעים באובייקט.
3. הגלים מוחזרים מהאובייקט כהדים.
4. המקלט מזהה את ההדים ומודד את הזמן שלוקח להם לחזור.



פורמט הפלט הוא אות דיגיטלי.

משמעותו של האות הדיגיטלי הוא זמן- מספר מיקרו השניות מאז שנשלח גל הקול עד שחזר ההד. לכן צריך לחשב את המרחק:

דרך=מהירות הקול באויר*הזמן שהתקבל/2/1,000,000

חיישן אור:

חיישן שמודד את עוצמת האור שנמצאת בסביבתו ומשנה את התנגדותו בהתאם.

הוא מורכב מדסקית עגולה העשויה מחומר קרמי המחולקת לשני אזורים מופרדים על ידי חומר הנקרא CdS.

חומר זה אחראי להתנהגות החשמלית של החיישן.

משני חצאי הקרמיים יוצאים שני קטבים שמתחברים למערכת.

חיישן אור הוא בעצם נגד משתנה, שמשתנה בהתאם לעוצמת האור אליו הוא חשוף.

ככול שעוצמת האור גדולה יותר ההתנגדות קטנה יותר.

הוא קולט את כמות עוצמת האור הנופלת על פני המשטח שלו.

איך עובד:

כאשר האור פוגע בCdS (החומר המוליך למחצה) הוא מעורר את האלקטרונים שבתוכו. זה מוביל לתנועה שלהם בתוך החומר כתוצאה מכך התנגדותו של החיישן יורדת.

פלט החיישן:

אות חשמלי/ אנלוגי.

הוא מייצג את התנגדות החיישן לאור שנופל עליו.

תווך הערכים המתקבלים: 0-1023

אם מעוניינים לקבל את הפלט באחוזים (ייצגו את אחוז החושך) נשתמש בפונקציה:

ערך המתקבל מהחיישן

idrvalue=map(ldrvalue,0,1023,0,100);

מצלמה:

המצלמה ESP32-CAM OV2460היא מודול מצלמה המבוסס על מיקרו-בקר ESP32 המשלב יכולות Wi-Fi ו-Bluetooth.

חלקי המצלמה והחיבורים:

חיישן המצלמה(OV2460)

החיישן אחראי על תפיסת התמונות והסרטונים. הוא ממיר את האור שנכנס דרך העדשה לאותות אלקטרוניים.

מיקרו-בקר ESP32

המיקרו-בקר מטפל בכל העיבוד של הנתונים המתקבלים מהחיישן ומנהל את התקשורת האלחוטית Wi-Fi ו Bluetooth.

חריץ לכרטיס MicroSD

מאפשר אחסון תמונות וסרטונים על כרטיס זיכרון.

חיבורי GPIO

מאפשרים חיבור רכיבים חיצוניים כמו חיישנים נוספים או פקדים.

מחבר USB

משמש לתכנות המודול דרך המחשב.

צילום תמונה:

תהליך הצילום מתחיל כאשר המיקרו-בקר ESP32 מקבל פקודת צילום. הקוד המופעל על המיקרו-בקר קורא לפונקציה esp_camera_fb_get() אשר אחראית על צילום התמונה.

החיישן (OV2460) בנוי ממטריצה של פיקסלים שרגישים לאור.

כאשר אור נופל על פיקסל, הוא מייצר מטען חשמלי פרופורציונלי לעוצמת האור. מטען זה מועבר כנתון דיגיטלי.

החיישן מקודד את האור הנקלט לכל פיקסל לתמונה דיגיטלית לרוב בפורמט RAW או YUV.

הנתונים הגולמיים נשלחים מהממשק הטורי של החיישן ל ESP32 באמצעות ממשק I2S או DVP.

המיקרו-בקר ESP32 מקבל את הנתונים הדיגיטליים מהחיישן דרך ממשק ה I2S

הנתונים הגולמיים נשמרים בזיכרון ה RAM של ה ESP32.

המיקרו-בקר מעבד את הנתונים הראשוניים כדי ליצור תמונה בפורמט שניתן להשתמש בו, כמו JPEG.

פונקציות של עיבוד תמונה כמו דחיסת JPEG מבוצעות על ידי הספרייה esp32-camera.

ה ESP32 משתמש באלגוריתמי עיבוד תמונה כדי לדחוס את התמונה לפורמט כמו JPEG כדי לחסוך מקום ולאפשר העברה מהירה דרך רשת.

שליחת הנתונים דרך Wi-Fi

לאחר עיבוד התמונה, ה ESP32 שולח את הנתונים דרך Wi-Fi למכשיר היעד.

המיקרו-בקר מגדיר שרת HTTP שיכול לקבל בקשות GET ולהחזיר את התמונה שצולמה.

הקוד עושה שימוש בספריות כמו WiFi וWebServer כדי להקים את השרת ולהגיב לבקשות.

שמירת הנתונים על כרטיס MicroSD

אם הקוד מתוכנת לכך, המיקרו-בקר יכול לשמור את התמונה על כרטיס ה MicroSD המחובר למודול.

<u>צילום וידאו:</u>

תהליך ההסרטה עם ESP32-CAM כולל תפיסת רצף של תמונות באופן רציף ושידורן או שמירתן כדי ליצור וידאו. התהליך מורכב מכמה שלבים מרכזיים, והוא דומה במובנים רבים

לתהליך צילום תמונה יחידה, אבל עם דגש על ביצוע חוזר ונשנה של השלבים בצורה מהירה ורציפה.

החיישן (OV2460) קולט פריימים (תמונות) באופן רציף.

כל פריים נתפס על ידי החיישן ומועבר למיקרו-בקר ESP32 כמו בצילום תמונה בודדת.

ספריות:

esp_camera.h- ספריה זו מספקת תמיכה ופונקציונליות לשימוש במצלמה מובנית. במקרה של ESP32 היא מאפשרת לקבל תמונות וסרטונים מהמצלמה ולנהל את ההגדרות שלה.

WiFi.h- ספריה זו מאפשרת למיקרו-בקר להתחבר ולנהל רשת WiFi . היא כוללת פונקציות להתחברות לרשת, ניהול חיבורים ושליטה בהגדרות רשת.

esp_timer.h- ספריה לניהול טיימרים ב ESP32. היא מספקת פונקציות ליצירת, קביעת, וניהול טיימרים בדיוק גבוה.

img_converters.h- ספריה להמרת תמונות בין פורמטים שונים, נגיד JPEG ל-BMP או לתבנית נתונים אחרת. זה עשוי להיות שימושי ביישומים שבהם נדרשת המרה של פורמטים תמונה.

fb_gfx.h- ספריה לציור על מסך, כולל תמיכה במגוון רחב של פונטים וצורות גרפיות על בסיס frame buffer מתאימה ליישומים בהם יש צורך בתצוגה גרפית פשוטה על מסך.

soc/soc.h ו**soc/rtc_cntl_reg.h,** -ספריות אלו מטפלות בבעיות של brownout שעלולות לקרות במקרה של מתח חשמלי נמוך ב ESP32. הן מספקות פונקציות לניהול ולהשבתת בעיות כאלה.

esp_http_server.h- ספריה זו מאפשרת ליצור שרת HTTP על ESP32 , מה שמאפשר למיקרו-בקר לשרת תוכן ולקבל בקשות HTTP ממכשירים אחרים ברשת.

מטרות ויעדים:

- חווית משתמש.

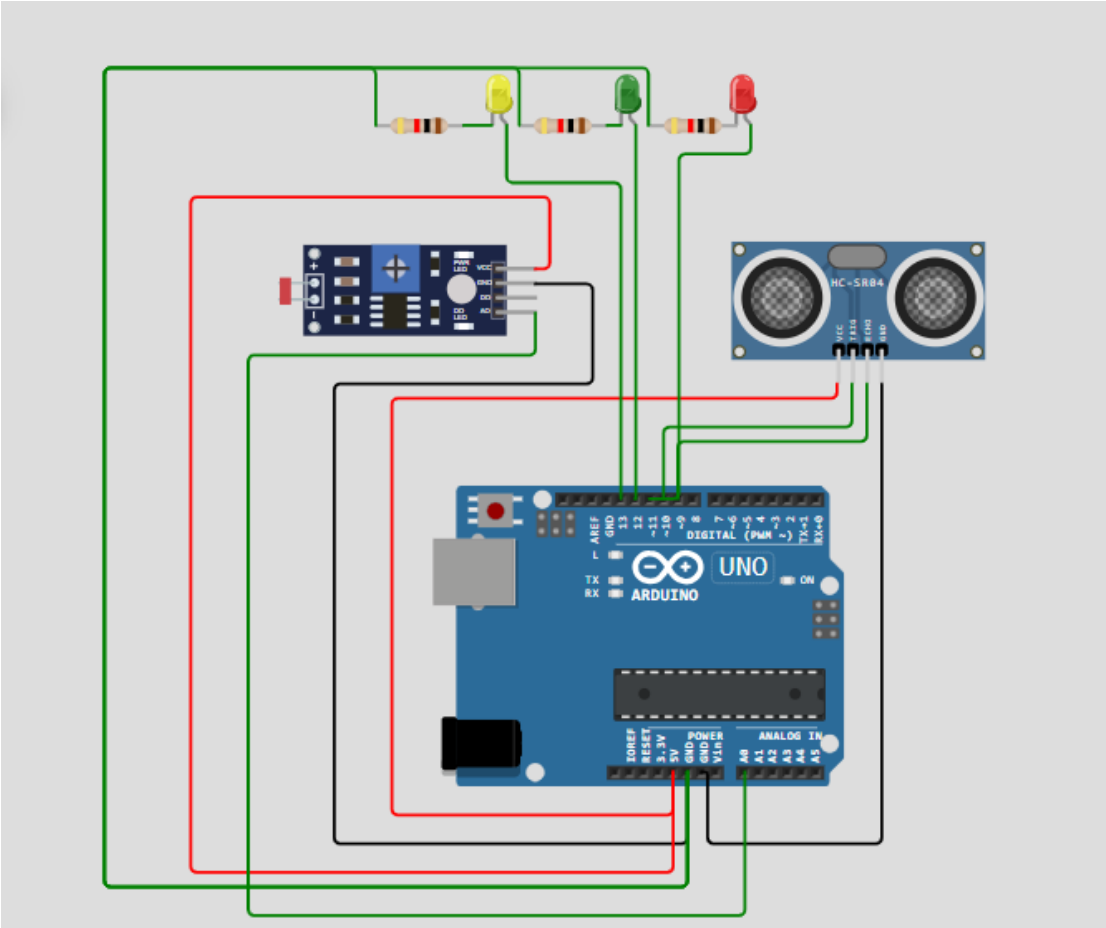
- תגובה במהירות מקסימלית.
- אבחון מדיוק ככל האפשר.

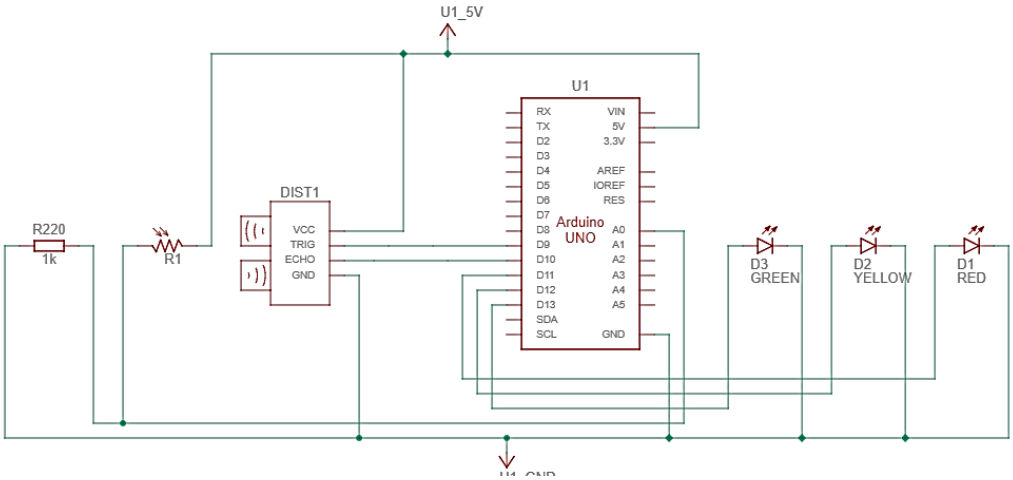
אתגרים:

- הפעלת חיישנים בצורה טובה.
- הפעלת מצלמה ושליחת נתונים ל #C.

תיעוד:

סרטוט חשמלי:





חיבורים:

חיישן אור:

V5 -VCC

0A -Analog output

GND -Gnd

חיישן מרחק:

V5 -VCC

9 -Trig

10 -Echo

GND -Gnd

מצלמה:

3.3V -3.3V

GND -GND

D2 -SDA

D1 -SCL

D5 -DC

D8 -CS

EN -RESET

D7 -MOSI

D6 -MISO

D4 -CLK

תיעוד קטעי קוד עיקריים:

Ultra.ino

```
// מגדיר קבוע שמציין את מספר הפין של הטריגר (שליחת אות)
#define PIN_TRIG 9
// מגדיר קבוע שמציין את מספר הפין של האקו (קבלת אות)
#define PIN_ECHO 10

// משתנה שישמור על משך הזמן הטוב
double good_duration;

void setup() {
  // מתחיל תקשורת סידורית בקצב 9600 ביט לשנייה
    Serial.begin(9600);
    // מגדיר את פין הטריגר ככלול הפלט (לשליחת אות)
    pinMode(PIN_TRIG, OUTPUT);
    // מגדיר את פין האקו ככלול הכניסה (לקבלת אות)
    pinMode(PIN_ECHO, INPUT);
    // משתנה פנימי חדש, אינו נחוץ ונמחק מיד אחרי סיום הפונקציה
    double good_duration;
}

bool check_distance() {
  // בודק אם משך הזמן הטוב נמצא בין 30 ל-60
    if (good_duration >= 30 && good_duration <= 60) {
      return true;
    } else {
      return false;
    }
}

void loop() {
    // הפעלת המשדר, התחלת מדידת זמן
    digitalWrite(PIN_TRIG, HIGH);
    delayMicroseconds(10);
    // שולח אות דיגיטלי נמוך דרך פין הטריגר
    digitalWrite(PIN_TRIG, LOW);

    // הפעלת המקלט, קריאת התוצאה
    int duration = pulseIn(PIN_ECHO, HIGH);
    // מחשב את משך הזמן הטוב בסמ לפי פולס אין
    good_duration = duration * 343.0 / 2 / 10000;
    delay(1000);
}
```

Ldr.ino

```
// כוללת את ספריית Arduino
#include <Arduino.h>

// מגדיר קבוע שמציין את הפין האנלוגי A0 לשימוש במקום LDR
const int LDR_PIN = A0;

// משתנה שיאחסן את הערך של ה-LDR
int ldrValue;

void setup() {
  Serial.begin(9600);
}

bool check_ldr() {
  // בודק אם ערך ה-LDR גדול מ-1000
  if (ldrValue > 1000) {
    return true;
  } else {
    return false;
  }
}

void loop() {
  // קורא ערך אנלוגי מהפין A0 (LDR_PIN) ומכניס אותו לתוך ldrValue
  ldrValue = analogRead(LDR_PIN);
  delay(1000);
}
```

camera.ino

```
"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" //disable brownout problems#
include "soc/rtc_cntl_reg.h"  //disable brownout problems#
"include "esp_http_server.h#

Replace with your network credentials//
;"const char* ssid = "Escolls_248641
;"const char* password = "12345678

"define PART_BOUNDARY "123456789000000000000987654321#

This project was tested with the AI Thinker Model, M5STACK PSRAM Model and M5STACK WITHOUT PSRAM //
define CAMERA_MODEL_AI_THINKER#
define CAMERA_MODEL_M5STACK_PSRAM#//
define CAMERA_MODEL_M5STACK_WITHOUT_PSRAM#//

Not tested with this model //
define CAMERA_MODEL_WROVER_KIT#//

if defined(CAMERA_MODEL_WROVER_KIT)#
define PWDN_GPIO_NUM    -1#
define RESET_GPIO_NUM   -1#
define XCLK_GPIO_NUM    21#
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      19#
define Y4_GPIO_NUM      18#
define Y3_GPIO_NUM       5#
define Y2_GPIO_NUM       4#
define VSYNC_GPIO_NUM   25#
define HREF_GPIO_NUM    23#
define PCLK_GPIO_NUM    22#

elif defined(CAMERA_MODEL_M5STACK_PSRAM)#
define PWDN_GPIO_NUM     -1#
define RESET_GPIO_NUM    15#
define XCLK_GPIO_NUM     27#
define SIOD_GPIO_NUM     25#
define SIOC_GPIO_NUM     23#

define Y9_GPIO_NUM       19#
define Y8_GPIO_NUM       36#
define Y7_GPIO_NUM       18#
define Y6_GPIO_NUM       39#
define Y5_GPIO_NUM        5#
define Y4_GPIO_NUM       34#
define Y3_GPIO_NUM       35#
define Y2_GPIO_NUM       32#
define VSYNC_GPIO_NUM    22#
define HREF_GPIO_NUM     26#
define PCLK_GPIO_NUM     21#

elif defined(CAMERA_MODEL_M5STACK_WITHOUT_PSRAM)#
define PWDN_GPIO_NUM     -1#
define RESET_GPIO_NUM    15#
define XCLK_GPIO_NUM     27#
define SIOD_GPIO_NUM     25#
define SIOC_GPIO_NUM     23#

define Y9_GPIO_NUM       19#
define Y8_GPIO_NUM       36#
define Y7_GPIO_NUM       18#
define Y6_GPIO_NUM       39#
define Y5_GPIO_NUM        5#
define Y4_GPIO_NUM       34#
define Y3_GPIO_NUM       35#
define Y2_GPIO_NUM       17#
define VSYNC_GPIO_NUM    22#
define HREF_GPIO_NUM     26#
define PCLK_GPIO_NUM     21#

elif defined(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#
else#
"error "Camera model not selected#
endif#
```

```
// קבועים לפרוטוקול ה-HTTP של הזרם החי מהמצלמה
;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

// משתנה גלובלי לניהול קישור ה-HTTP
;httpd_handle_t stream_httpd = NULL

// פונקציה לטיפול בבקשות HTTP לשרת המצלמה
}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
;[64]char * part_buf

// הגדרת סוג התוכן לפי הפרוטוקול המבוקש
;res = httpd_resp_set_type(req, _STREAM_CONTENT_TYPE)
}if(res != ESP_OK)
;return res
{

// לולאה אינסופית לקבלת תמונות מהמצלמה ושליחתן כפרקים בפרוטוקול ה-HTTP
}while(true)
;()fb = esp_camera_fb_get
} if (!fb)
;Serial.println("Camera capture failed")
;res = ESP_FAIL
} else {
// בדיקה אם צריך להמיר ל-JPEG או שהתמונה כבר בפורמט זה
}if(fb->width > 400)
}if(fb->format != PIXFORMAT_JPEG)
;bool jpeg_converted = frame2jpg(fb, 80, &_jpg_buf, &_jpg_buf_len)
;esp_camera_fb_return(fb)
;fb = NULL
}if(!jpeg_converted)
;Serial.println("JPEG compression failed")
;res = ESP_FAIL
{
} else {
;jpg_buf_len = fb->len_
;jpg_buf = fb->buf_
{
{
{
// שליחת פרק ה-HTTP בפורמט multipart
}if(res == ESP_OK)
;size_t hlen = snprintf((char *)part_buf, 64, _STREAM_PART, _jpg_buf_len)
;res = httpd_resp_send_chunk(req, (const char *)part_buf, hlen)
{
}if(res == ESP_OK)
;res = httpd_resp_send_chunk(req, (const char *)_jpg_buf, _jpg_buf_len)
{
}if(res == ESP_OK)
;res = httpd_resp_send_chunk(req, _STREAM_BOUNDARY, strlen(_STREAM_BOUNDARY))
{
// סיום טיפול במצלמה ובזיכרון כאשר הגיע לסיום התמונה הנוכחית
}if(fb)
;esp_camera_fb_return(fb)
;fb = NULL
;jpg_buf = NULL_
}(else if(_jpg_buf {
;free(_jpg_buf)
;jpg_buf = NULL_
{
// בדיקת תקינות הפעולה ויציאה מהלולאה אם קרתה שגיאה
}if(res != ESP_OK)
;break
{
// פלט לכימות הדקות
```

```
;Serial.printf("MJPG: %uB\n",(uint32_t)(_jpg_buf_len))//
{
;return res
{

// פונקציה להתחלת שרת ה-HTTP לשידור התמונות
}()void startCameraServer
;()httpd_config_t config = HTTPD_DEFAULT_CONFIG
;config.server_port = 80

// הגדרת אינדקס URI עבור פונקצית הטיפול בזרימת התמונות
} = httpd_uri_t index_uri
,"/" =      uri.
,method    = HTTP_GET.
,handler   = stream_handler.
user_ctx  = NULL.
;{

;Serial.printf("Starting web server on port: '%d'\n", config.server_port)//
// התחלת השרת ה-HTTP ורישום האינדקס URI
} if (httpd_start(&stream_httpd, &config) == ESP_OK)
;httpd_register_uri_handler(stream_httpd, &index_uri)
{
{

// הגדרות ראשיות והתחלת התוכנית
} ()void setup
WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout detector

;(115200)Serial.begin
;Serial.setDebugOutput(false)

// הגדרות המצלמה
;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

// הגדרת המצלמה עם הגדרות מתאימות לפי קיומו של PSRAM
}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
{

// אתחול המצלמה ובדיקת תקינות
;esp_err_t err = esp_camera_init(&config)
} if (err != ESP_OK)
;Serial.printf("Camera init failed with error 0x%x", err)
```

```
;return
{

// התחברות לרשת WiFi
;WiFi.begin(ssid, password)
} while (WiFi.status() != WL_CONNECTED)
;(500)delay
;(".")Serial.print
{
;("")Serial.println
;Serial.println("WiFi connected")

// הדפסת כתובת IP לצורך גישה למצלמה
;Serial.print("Camera Stream Ready! Go to: http://")
;Serial.print(WiFi.localIP())

// התחלת שרת ה-HTTP לשידור התמונות
;()startCameraServer
{

// לולאת הרצת התוכנית, ריקה מכיוון שהקוד מבוסס על אירועים ולא זקוק ללולאת תקשורת פעילה
} ()void loop
;(1)delay
```

שרת C#:

Diagnosis/Tounge:

פונקציות עזר לפונקציה Number_out:

Run_YOLO:

הפונקציה מריצה פקודת קומנד שמחלצת תגיות לקובץ טקסט עבור סרטון. הפונקציה יוצרת תהליך להרצת פקודת קומנד ומריצה אותו. כדי שהפקודה תצליח התהליך צריך לישון שניה כדי שלא יסגר לפני הרצת הפקודה.

```
public static void Run_YOLO()
{
    string command = @"python detect.py --weight runs/train/Model6/weights/last.pt --img-size 640
                       --source C:\Users\user1\Documents\PROJECT\TextAnalysis\video.mp4
                        >> C:\Users\user1\Documents\PROJECT\TextAnalysis\results.txt 2>&1";
    string path = @"C:\Users\user1\yolov5-master";

    Process process = new Process();
    ProcessStartInfo startInfo = new ProcessStartInfo
    {
        WorkingDirectory = path,
        FileName = "cmd.exe",
        RedirectStandardInput = true,
        RedirectStandardOutput = true, // קבלת פלט מהפקודה
        UseShellExecute = false,
        CreateNoWindow = true
    };

    process.StartInfo = startInfo;
    process.Start();

    process.StandardInput.WriteLine(command);
    Thread.Sleep(1000);

    process.Close();
}
```

Make_Auto:

הפונקציה בונה אוטומט מילים למילים in ו- out ומערך מצבים סופיים כך שלמילה in המצב הסופי יהיה מחרוזת עם המספר 1, למילה out יהיה מחרוזת עם המספר 0 ושאר המילים מחרוזת ריקה.

```
public static void Make_Auto()
{
    auto = new int[5, 26];
    auto[0, Hash('i')] = 1;
    auto[1, Hash('n')] = 2;
    auto[0, Hash('o')] = 3;
    auto[3, Hash('u')] = 4;
    auto[4, Hash('t')] = 5;
    string[] final_state1 = { "", "", "1", "", "", "0"};
    final_state = final_state1;
}
```

הפונקציה משתמשת בפונקציית הגיבוב Hash שמחזירה מספר סידורי עבור כל אות מה -a b האנגלי.

```
public static int Hash(char t)//מחזירה ערך מספרי עבור כל אות- 1 ל- a, 2 ל- b וכו.
{
    return (int)t - (int)'a';
}
```

In_or_out:

הפונקציה מקבלת מילה ומחזירה מצב סופי שלה(מחרוזת כלשהי) לפי האוטומט.

```
public static string In_or_out(string word)
{
    string tav;
    try {
    char[] text = word.ToCharArray();
    int current_status = 0;
    for (int i = 0; i < text.Length; i++)
    {
        current_status = auto[current_status, Hash(text[i])];
    }
    tav=final_state[current_status];
    }
    catch {
        tav = "";
    }
    return tav;
}
```

Number_out:

הפונקציה מחזירה את מספר הפעמים שהלשון זוהתה מחוץ לפה במהלך הסרטון. לאחר הרצת היולו באמצעות זימון הפונקציה Run_YOLO והכנת האוטומט באמצעות זימון הפונקציה Make_Auto, הפונקציה קוראת את קובץ הטקסט של תוצאות היולו למשתנה מחרוזת כאשר פקודת הקומנד סיימה לרוץ והקובץ ניתן לקריאה. הפונקציה מחלקת את תוכן הקובץ למילים, עוברת בלולאה על כל המילים ושולחת כל מילה לפונקציה In_or_out לקבלת מצב סופי ומוסיפה את מה שחזר למשתנה מחרוזת str_bits. לאחר מכן הפונקצציה מכינה מחרוזת ביטים חדשה טובה. כל ביט i במחרוזת new_ str_bits יהיה הביט הרווח מבין הביטים i, i+1, i+2 במחרוזת str_bits.

המערך _0_or_1 מאותחל כך:

```
public static int[] _0_or_1 = {0,0,1,1};
```

לאחר סידור המחרוזת ביטים הפונקציה עוברת בלולאה על המחרוזת new_ str_bits ומונה את הפעמים בהם יש רצף "01" שזה אומר שהלשון יצאה.

```
public static int Number_out()
{
    Run_YOLO();
    Make_Auto();
    int mone = 0;
    bool degel = false;
    while (degel == false)
    {
        try
        {
            string file = File.ReadAllText("C:/Users/user1/Documents/PROJECT/TextAnalysis/results.txt");
            degel = true;
            string str_bits = "";
            string[] words = Regex.Split(file, @"\W+");
            string bit;

            for (int i = 1; i < words.Length - 1; i++)//make str of bits of all frames
            {
                bit = In_or_out(words[i]);
                str_bits+= bit;
            }

            string new_str_bits = "";
            int index;
            for (int i=0; i<str_bits.Length; i++)//make new str of good bits
            {
                index = int.Parse(str_bits[i].ToString())+ int.Parse(str_bits[i+1].ToString())+ int.Parse(str_bits[i+2].ToString());
                new_str_bits += _0_or_1[index];
            }

            mone = int.Parse(new_str_bits[0].ToString());
            for (int i = 0; i < new_str_bits.Length; i++)//number str '01'- the tounge out
            {
                index = int.Parse(new_str_bits[i].ToString())- int.Parse(new_str_bits[i+1].ToString());
                if (index == -1)//if have 0 befor 1
                    mone++;
            }


            File.Delete("C:/Users/user1/Documents/PROJECT/TextAnalysis/results.txt");
            File.Delete("C:/Users/user1/Documents/PROJECT/TextAnalysis/video.mp4");
        }
        catch { }
    }

    return mone;
}
{
```

תיאור הארכיטקטורה:

הארכיטקטורה של הפתרון המוצע בפורמט של Top-Down level Design:

SERVER-C#

BLL

Tounge.cs- בקובץ זה כתובות הפונקציות של האבחון.

API

Connect.cs- קובץ שבו קיים החיבור לשרת מצלמה וזימון לפונקציה הראשית של האבחון.

CLIENT- ARDUINO UNO

מכשיר הכולל:

- חיישן אור- LDR
- חיישן מרחק אולטרסוני

Ultra.ino- קובץ שבו הפונקציה שמתעסקת עם החיישן אור.

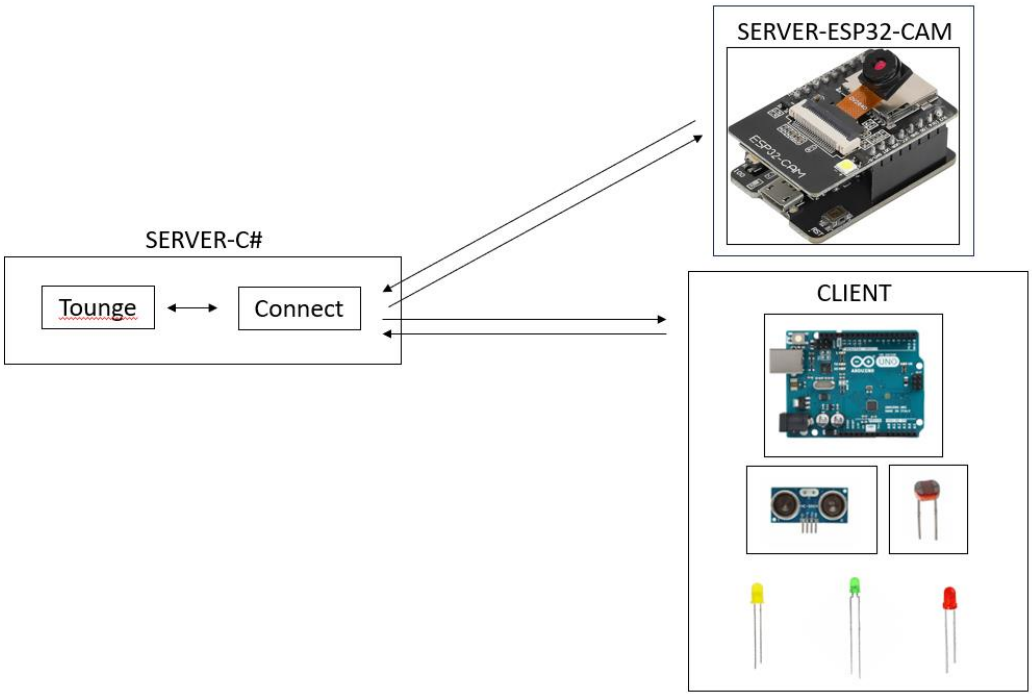
Ldr.ino- קובץ שבו הפונקציה שמתעסקת עם החיישן מרחק.

CAM -SERVER

- מצלמה- ESP32-CAM OV2640

Camera.ino- קובץ שמפעיל את המצלמה ושולח את הפריימים ל C#.

תיאור הרכיבים בפתרון:



תיאור פרוטוקולי התקשורת:

I2C:

I2C הוא פרוטוקול תקשורת טורית סינכרוני שפותח על ידי פיליפס (כיום NXP Semiconductors)
בתחילת שנות ה-80.

הוא משמש לתקשורת בין התקנים שונים בלוח מעגלים או בתוך מערכת.

I2C הוא פרוטוקול תקשורת מאסטר-עבד, כלומר יש התקן אחד (המאסטר) שיוזם תקשורת עם מכשירים אחרים
(העבדים) באותו אפיק.

ההתקן הראשי שולח נתונים להתקן העבד ומקבל נתונים מהתקן העבד.

התקן העבד מגיב רק לפקודות מהמכשיר הראשי.

פרוטוקול I2C משתמש בשני חוטים, הנקראים קווי SDA (נתונים טוריים) ו-SCL (שעון טורי),
כדי להעביר נתונים בין המכשירים.

קו SDA משמש להעברת נתונים מהמאסטר לעבד, ולהיפך.

קו SCL משמש לסנכרון התקשורת בין המכשירים.

פרוטוקול I2C משתמש בשיטת התחלה-עצירה של תקשורת.

לפני שהמכשיר הראשי יכול לתקשר עם התקן עבד, עליו לשלוח תחילה אות התחלה כדי לציין את תחילת השידור.

אות ההתחלה הוא מעבר נמוך לגבוה בקו SDA בעוד קו SCL גבוה.

לאחר שליחת אות ההתחלה, המכשיר הראשי שולח את הכתובת של מכשיר העבד איתו הוא רוצה לתקשר.

הכתובת היא בדרך כלל באורך של 7 ביטים, אך ניתן להרחיב אותה ל-10 ביטים עבור מערכות גדולות יותר.

מכשיר העבד עם הכתובת המתאימה יגיב באות אישור.

לאחר שהתקן העבד אישר את ההתקן הראשי, המאסטר יכול לשלוח או לקבל נתונים מהעבד.

המאסטר שולח נתונים על קו SDA בעוד קו SCL גבוה. התקן העבד מקבל את הנתונים ושולח אות אישור למכשיר הראשי.

כאשר המכשיר הראשי סיים לשלוח או לקבל נתונים, הוא שולח אות עצור כדי לציין את סיום השידור.

אות העצירה הוא מעבר גבוה לנמוך בקו SDA בעוד שקו SCL גבוה.

http

פרוטוקול HTTP הוא פרוטוקול תקשורת מבוסס טקסט המשמש להעברת מידע בין רכיבי רשת באינטרנט.

פרוטוקול HTTP שרת-לקוח:

בפרוטוקול HTTP בסוג זה, יש שני סוגי פעולות: בקשות ותגובות.

בקשות (Requests): הן הודעות שנשלחות מהלקוח לשרת בכדי לבקש מידע או לבצע פעולה מסוימת. בקשת HTTP מכילה מספר ראשים (headers) שמציינים את סוג הבקשה (GET, POST, PUT, DELETE ,POST)

תגובות (Responses): הן הודעות שנשלחות מהשרת ללקוח כתגובה לבקשה שנשלחה. תגובת HTTP מכילה גם ראשים (headers) שמציינים את מצב התגובה קוד סטטוס וגוף התגובה שיכול להיות מידע מבוקש, תמונה, קובץ וכדומה.

פרוטוקול HTTP שרת-שרת:

בפרוטוקול זה, שני שרתים משתפים מידע באמצעות HTTP. השימוש הנפוץ ביותר הוא בשירותי API ושירותי ענן. כאשר קיימים שני שרתים כאלה, הם עשויים לתקשר עם זה את הזמן באמצעות הצגת שאילתות אחת לשנייה כיצד מתקיים

שרת- לקוח:

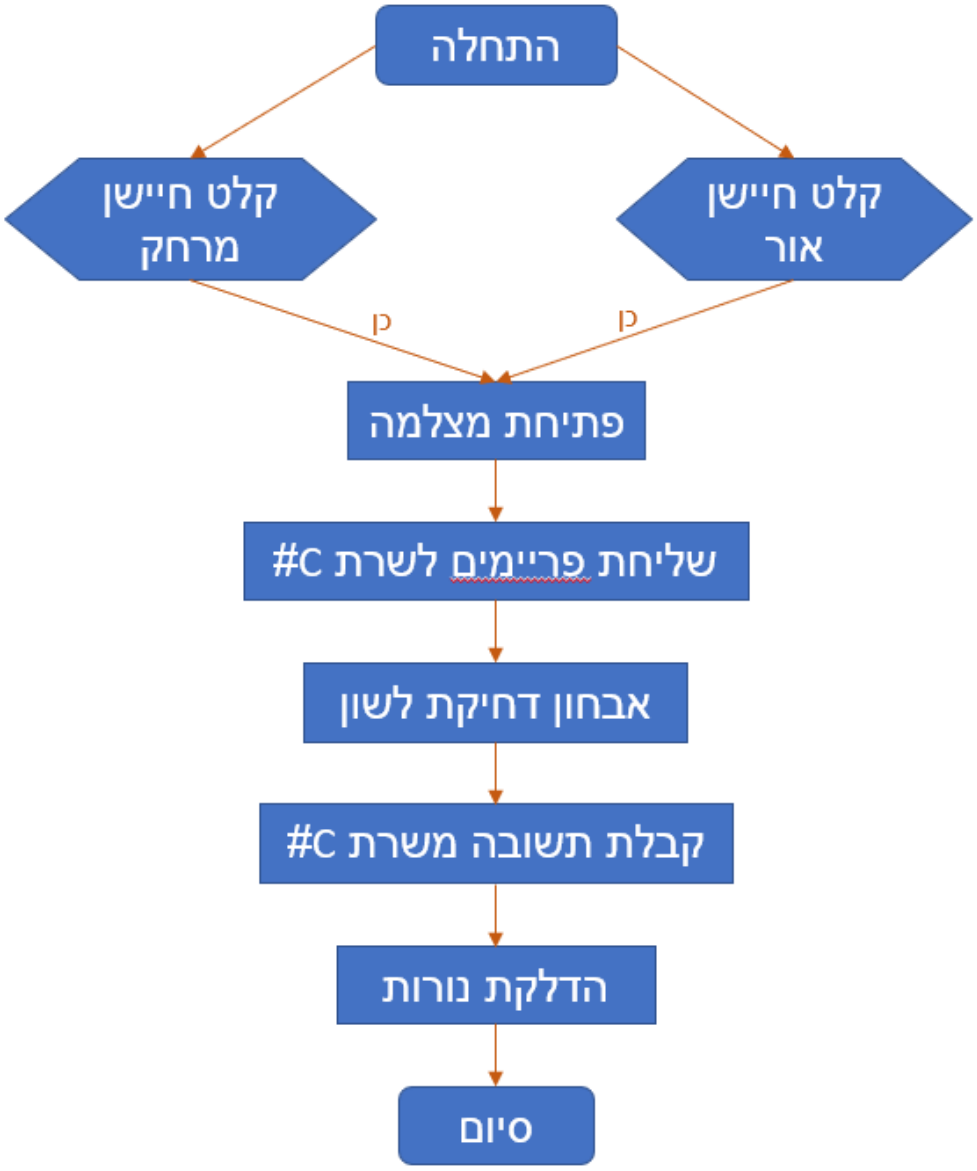
צד שרת: ESP32-CAM

צד שרת: #C

צד לקוח- ARDUINO UNO

תרשימים:

תרשים זרימה:



תרשים פונקציונאלי:

bool check_ldr()- פונקציה שבודקת אם יש מספיק תאורה בחדר.

bool check_distance()- פונקציה שבודקת אם המרחק של העצם בין 30 ל- 50 ס"מ.

static esp_err_t stream_handler(httpd_req_t *req)- פונקציה זו מטפלת בבקשות HTTP לצורך הזרמת התמונות מהמצלמה ומשתמשת בפונקציות מערכת המצלמה כדי לקבל

את התמונה הנוכחית, להמיר אותה ל JPEG ולשלוח אותה חזרה ללקוח באמצעות תגובה רציפה.

void startCameraServer()- פונקציה זו מגדירה ומפעילה את השרת ה- HTTP שמאזין לבקשות להזרמת התמונות.

void setup()- בפונקציה זו מתבצעים הגדרות ראשוניות של המצלמה, אתחול של ה- WiFi עם שם הרשת והסיסמה, והתחלת השרת ה- HTTP להזרמת התמונות.

void loop()- בלולאה זו נשמרת התמונה הנוכחית ונוספת לזרימה הראשית של התמונות שמשודרות על ידי השרת ה- HTTP.

מדריך למשתמש:

עמוד מול המצלמה במרחק שבין 30- 60 ס"מ, עם תנאי תאורה טובים בחדר.

לחץ על הכפתור של פתיחת מצלמה, דבר מול המצלמה, לחץ לסיום והמתן להדלקת נורה.

אדום מהבהב- דחיקת לשון קשה

אדום- דחיקת לשון בינונית

ירוק- דחיקת לשון קלה

צהוב- אין דחיקת לשון

במקרה והמצלמה לא נפתחת נסה להתקרב או להתרחק ממנה, או להגביר את התאורה בחדר.

פיתוחים עתידיים:

- הוספת מסך עם מסרים מפורטים אודות האבחון ובעיות במרחק ובתאורה.
- מסרים בזמן אמת- ברגע שהלשון יוצאת, תהבהב נורה אדומה.

סיכום ומסקנות

לאחר חודשים של עמל, הגעתי אל היעד, יצרתי בכוחות עצמי מכשיר אלקטרוני שמייצג כמעט את הפרויקט גמר שעשיתי, שבשלבי הדמיון לא כ"כ האמנתי בו אבל קיוויתי שלפחות חלק אצליח להשיג. הדרך לא הייתה קלה בכלל, ניסיתי שוב ושוב עד שהצלחתי.

למדתי לעבוד ולהתמודד לבד עם דברים חדשים, להנות מלימוד עצמאי.

דבר נוסף שהקשה עלי מאוד, אך התמודדתי איתו יפה הוא העבודה המסודרת שנדרשת שלב אחר שלב. אני אוהבת לעשות דברים בגדול, לדעת שסיימתי ומאוחר יותר לתקן אותם. בעבודה בפרויקט לאחר שחוויתי את ההשלכות של עבודה לא מושלמת הגעתי למסקנה שלא עוברים שלב בלי שמסיימים שלב קודם בצורה פרפקציוניסטית

אין ספק שהידע שרכשתי במהלך הפרויקט הוא גדול ושווה כל מאמץ.

ביבליוגרפיה:

- Analyticsvidhya
- Makesense
- Stackoverflow
- Random nerd tutorials
- wokwi
- arduino