**Bitácora personal**

**Granata Joaquín – 7° 2° Aviónica Comisión A – Proyecto: VIA**

11/4 (viernes 11 de abril)

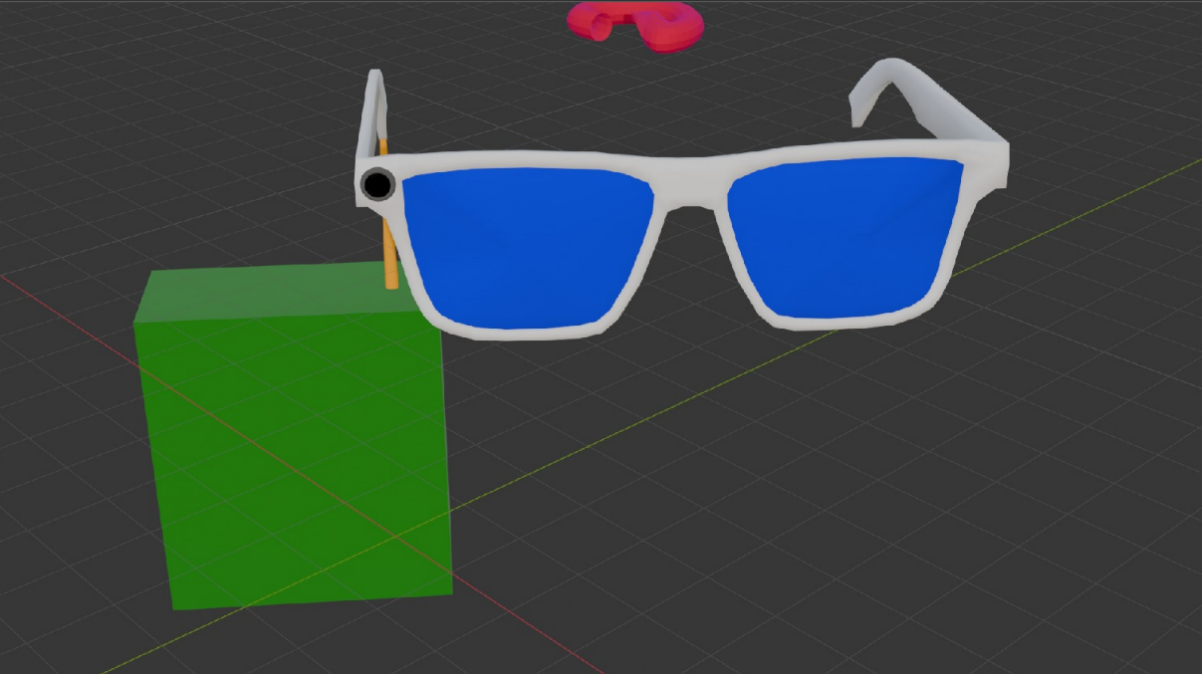
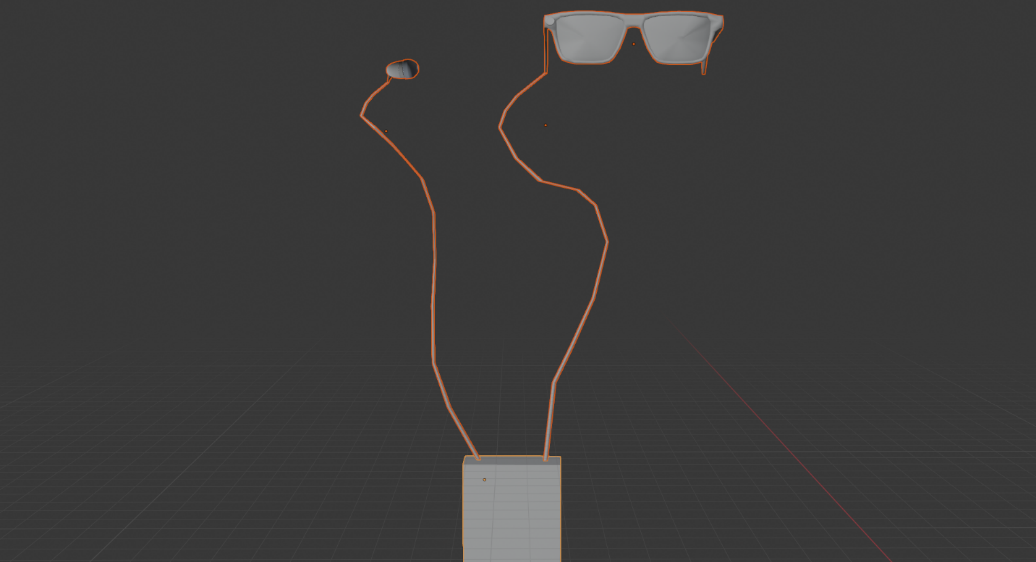
Empiezo un modelo del proyecto en Blender.

Empiezo a ver tutoriales para aprender Blender.

16/4 (miércoles 16 de abril)

Continuo con el modelaje del proyecto armado y con el aprendizaje de Blender

Avance hasta la fecha



22/4 (martes 22 de abril)

Luego de definir nuestros puestos, me dedico a aprender como programar una pagina

23/4 (miércoles 23 de abril)

Comienzo con el aprendizaje y programación de la App. Primero, empezare programando una versión de página de la App para luego pasar a una App

29/4 (martes 29 de abril)

Continuo con el aprendizaje de programación

Configuro botones para que redirijan a Maps.

6/5 (martes 6 de mayo)

Continuo con el aprendizaje de programación

Consigo que la página pida el nombre y datos del acompañante o persona de confianza al que mandar la ubicación o datos de confianza

9/5 (viernes 9 de mayo)

Nos comunicamos con una organización de ayuda a la gente no vidente para acordar una reunión. Esto con el fin de obtener su apoyo.

13/5 (martes 13 de mayo)

Ayudo con el armado y ensayo de sensor de proximidad con circuito en Wokwi. Debido a problemas con la simulación, se tuvo que hacer la programación y simulación con Arduino.



#define TRIG\_PIN 9

#define ECHO\_PIN 8

#define BUZZER\_A 7

#define BUZZER\_B 6

#define BUZZER\_C 5

void setup() {

**Serial**.begin(9600);

  pinMode(TRIG\_PIN, OUTPUT);

  pinMode(ECHO\_PIN, INPUT);

  pinMode(BUZZER\_A, OUTPUT);

  pinMode(BUZZER\_B, OUTPUT);

  pinMode(BUZZER\_C, OUTPUT);

}

void loop() {

  digitalWrite(TRIG\_PIN, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN, LOW);

  long duracion = pulseIn(ECHO\_PIN, HIGH);

  float distancia = duracion \* 0.0343 / 2;

  // Apaga todos los buzzers antes de activar el correcto

  digitalWrite(BUZZER\_A, LOW);

  digitalWrite(BUZZER\_B, LOW);

  digitalWrite(BUZZER\_C, LOW);

  if (distancia <= 200) {

**Serial**.println("Objeto adelante");

    digitalWrite(BUZZER\_A, HIGH);

  }

  else if (distancia > 200 && distancia <= 350) {

**Serial**.println("Objeto acercándose");

    digitalWrite(BUZZER\_B, HIGH);

  }

  else {

**Serial**.println("Camino despejado");

    digitalWrite(BUZZER\_C, HIGH);

  }

  delay(1000);

}

30/5 (viernes 30 de mayo)

Realizamos una reunión con la directora de la organización ASAC, una organización que ayuda a la gente no vidente. La razón de la reunión fue la de recopilar y reunir información sobre algunos de los problemas que la gente no vidente debe afrontar en su día a día para poder implementarla en distintos aspectos de nuestro proyecto. Además, acordamos vernos de nuevo para que nos den su feedback de futuras actualizaciones o avances.



3/8 (martes 3 de junio)

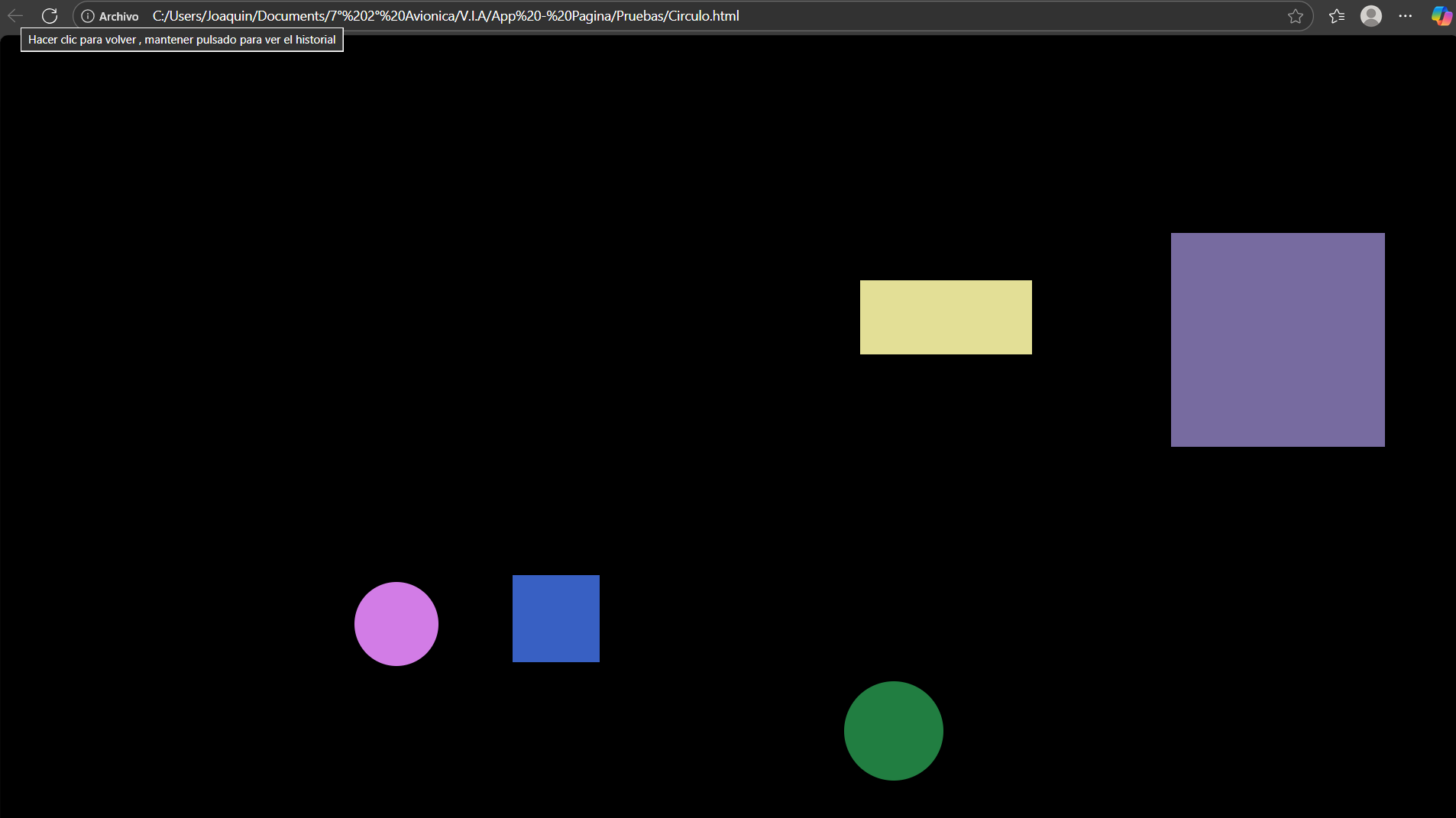
Primera vez en Newton. Trabajamos en lo que nos piden

4/6 (miércoles 4 de junio)

Empiezo a aprender a programar una aplicación para el proyecto. La app tendrá todas las funciones que la pagina

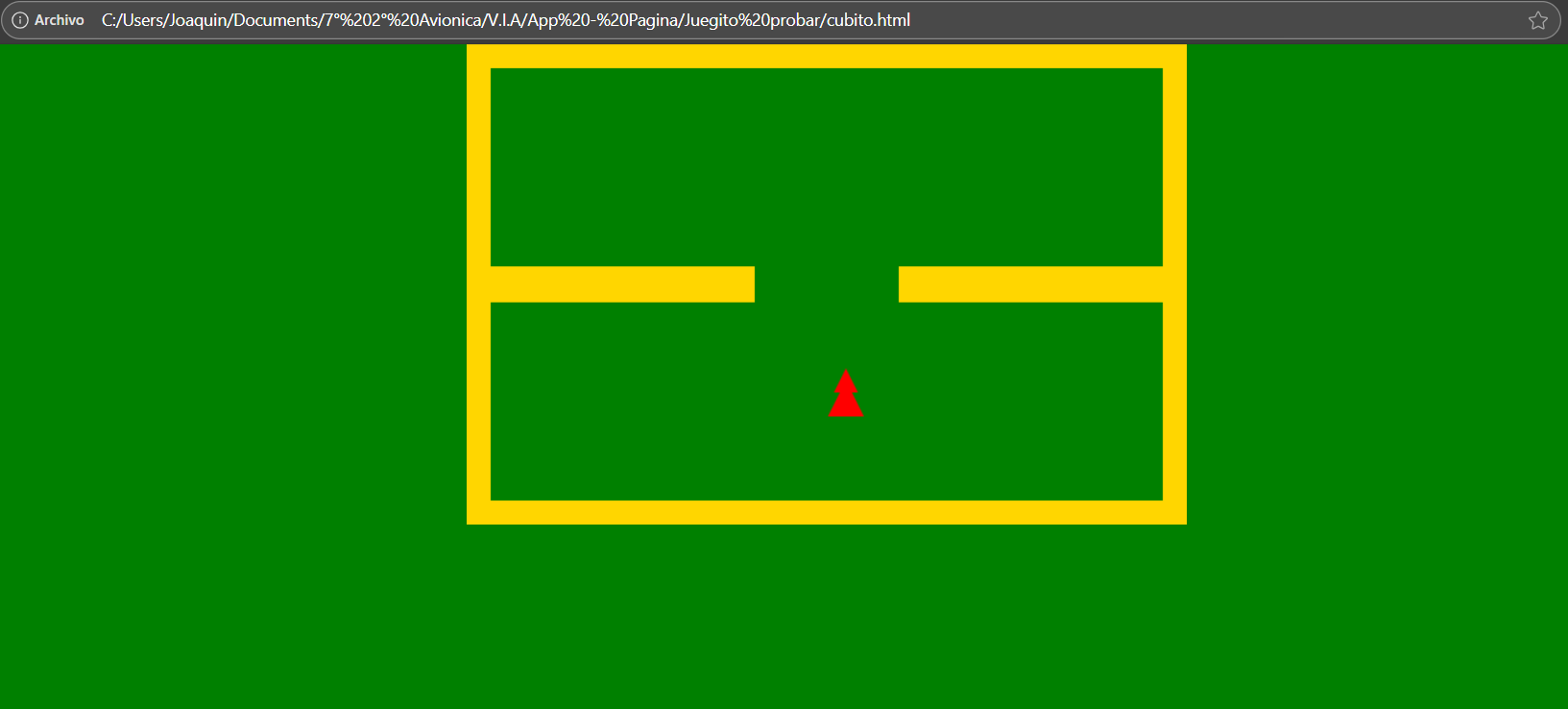
10/6 (martes 10 de junio)

Armo una versión de página para empezar. Arranco probando las bases de la programación de aplicaciones



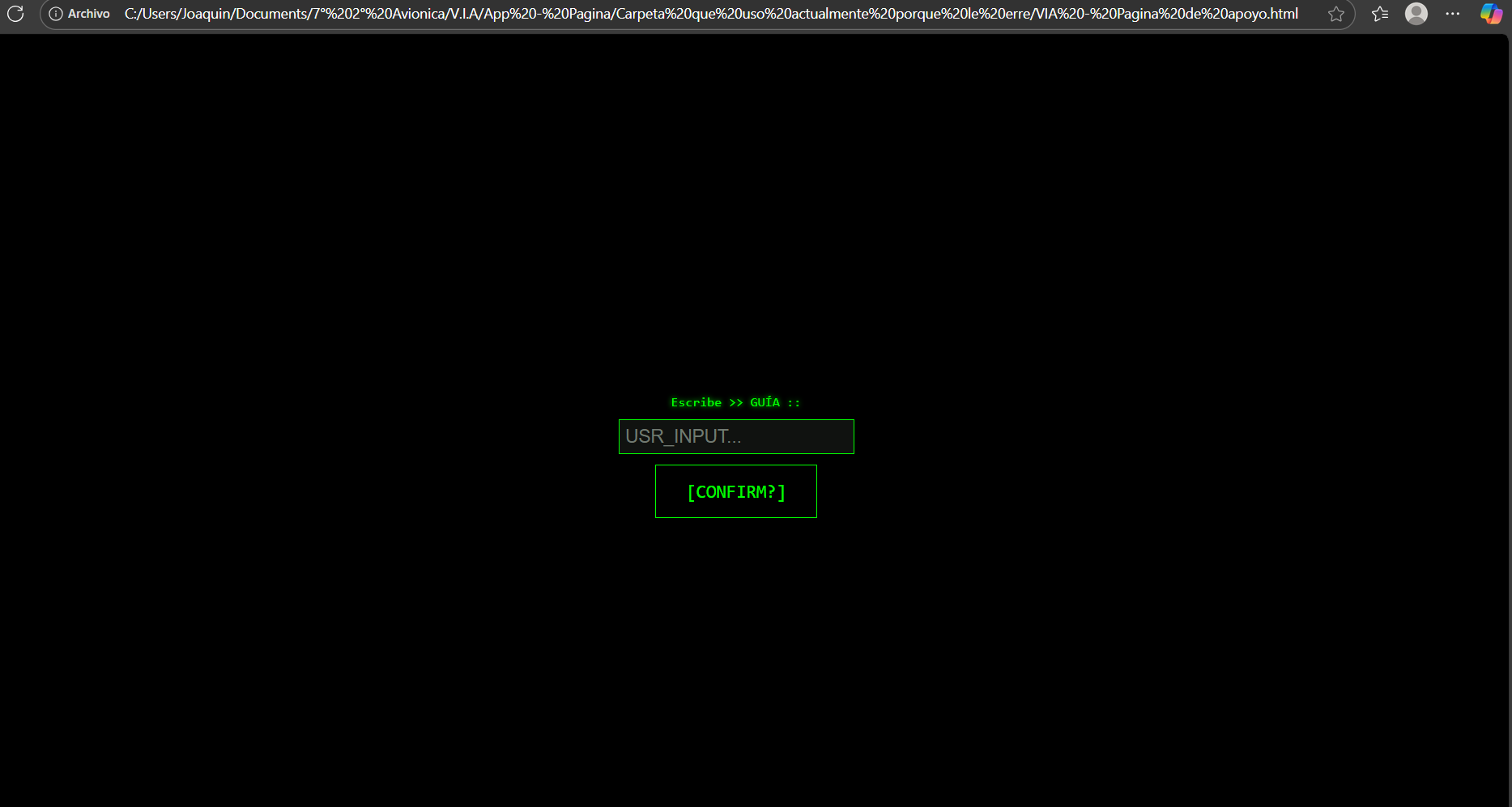
11/6 (miércoles 11 de junio)

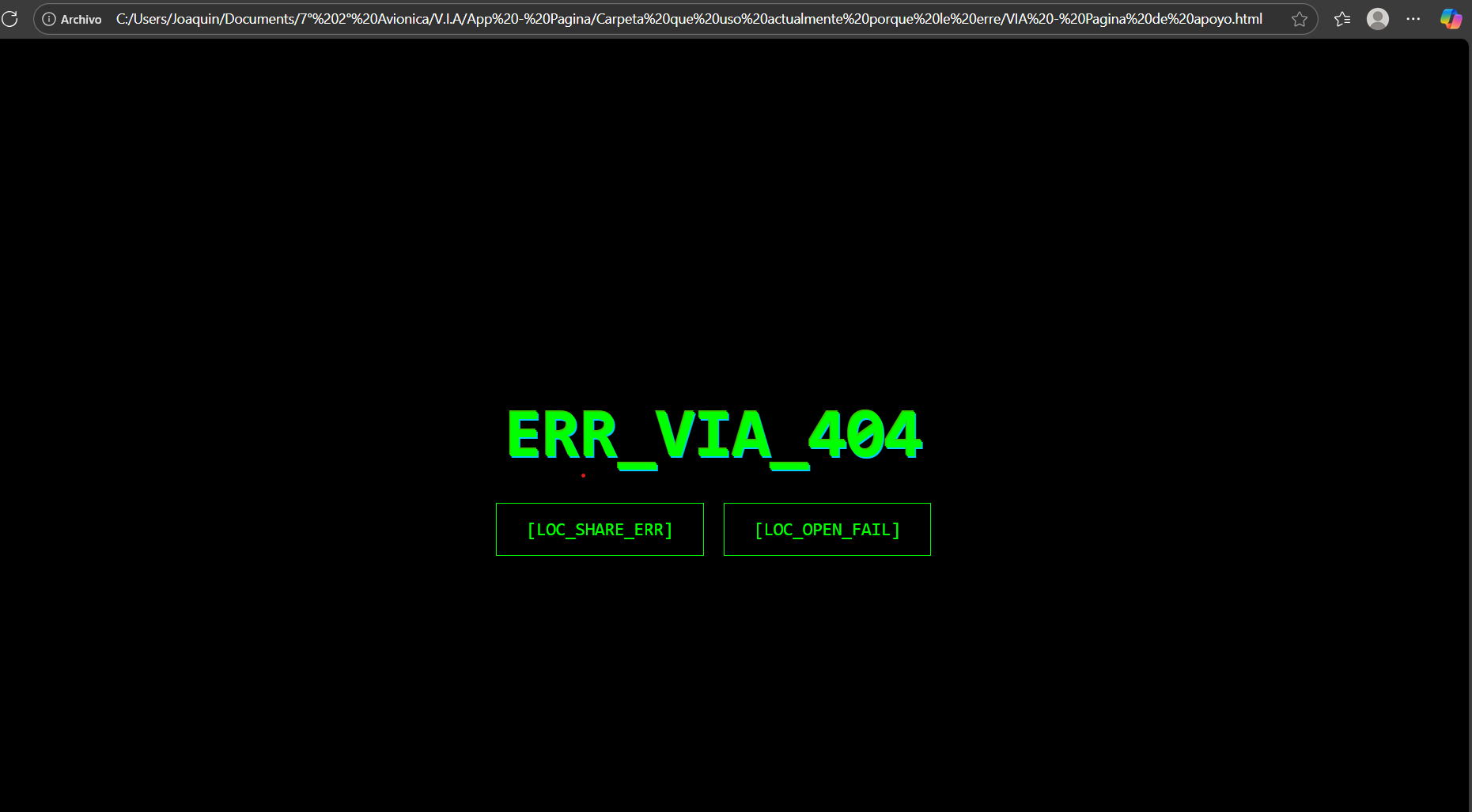
Aprendo a generar formas específicas, cambiar el fondo, y generar bordes y contrastes de distintos colores



17/6 (martes 17 de junio)

Empiezo una prueba de la página. La idea principal es que el usuario ponga el nombre y contacto de su compañero o persona de confianza. Después, tendrá la opción de enviar su ubicación a esa persona designada o puede recibir, mediante un audio, le diga donde se encuentra.

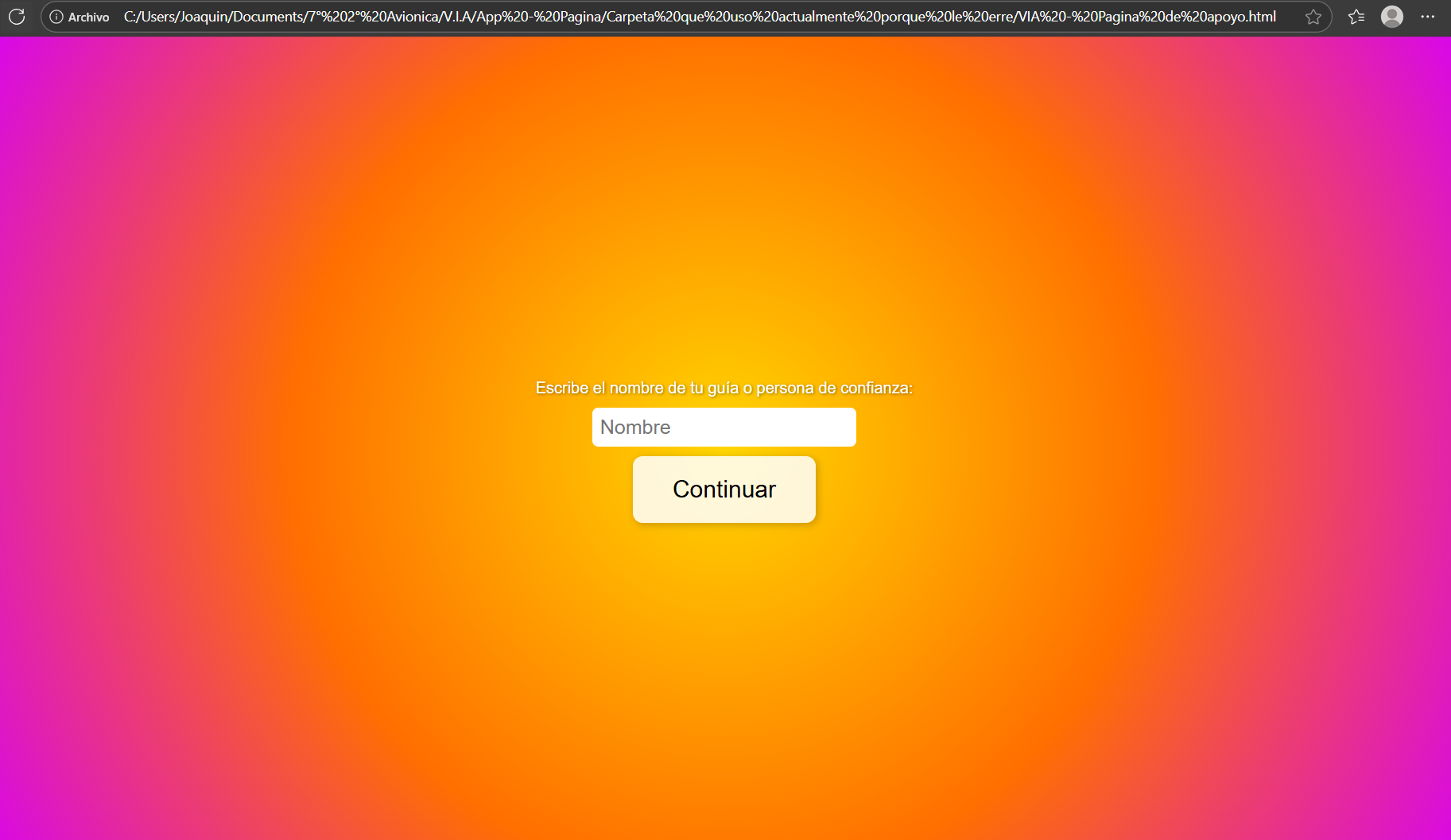




El inicio fue medio, complicado. Se me presentan varios problemas, aunque el más común es uno donde los textos no se muestran adecuadamente o directamente no se muestran directamente. En estos ejemplos, la página no cargo correctamente los textos, por lo que mostraba los textos que definí como “Erróneo”. A su vez, los colores no se mostraron, excepto por el de los textos. Intentare solucionarlo durante estos días.

24/6 (martes 24 de junio)

Logro arreglar los errores y añado nuevos detalles y colores.





Ahora, debo lograr que los botones abran Google Maps y realicen sus respectivas funciones.

25/6 (miércoles 25 de junio)

Logro que los botones abran Google Maps. Sigo intentando lograr que cada uno realice su tarea designada

1/7 (martes 1 de julio)

Comienzo a trabajar con la Raspberry y la ESP32-CAM para hacer el video y el reconocimiento. Voy a empezar con un código simple para probar el funcionamiento de la cámara.

2/7 (miércoles 2 de julio)

Después de mucha prueba y error, consigo que cada botón realice sus cometidos. Aun así, sigo trabajando para mejorar la app y añadir nuevas mecánicas.

18/7 (viernes 18 de julio)

Pruebo el código del ESP32.

La primera versión indicaba si la cámara estaba conectada o no. Aun así, debido a la manera en la que se conecta la ESP32, se debe usar un transformador de TTL a Serial.

21/7 – 4/8 (21 de julio al 4 de agosto)

**¡¡VACACIONES DE INVIERNO!!**

5/8 (martes 5 de agosto)

Para una mejor comunicación, me dedico a programar una función de la página donde, tanto el usuario como su persona de confianza añaden sus mails y nombres.

Esto con el fin de que, cuando se pida Compartir ubicación, le llegue a la persona de confianza un mensaje a su Gmail.

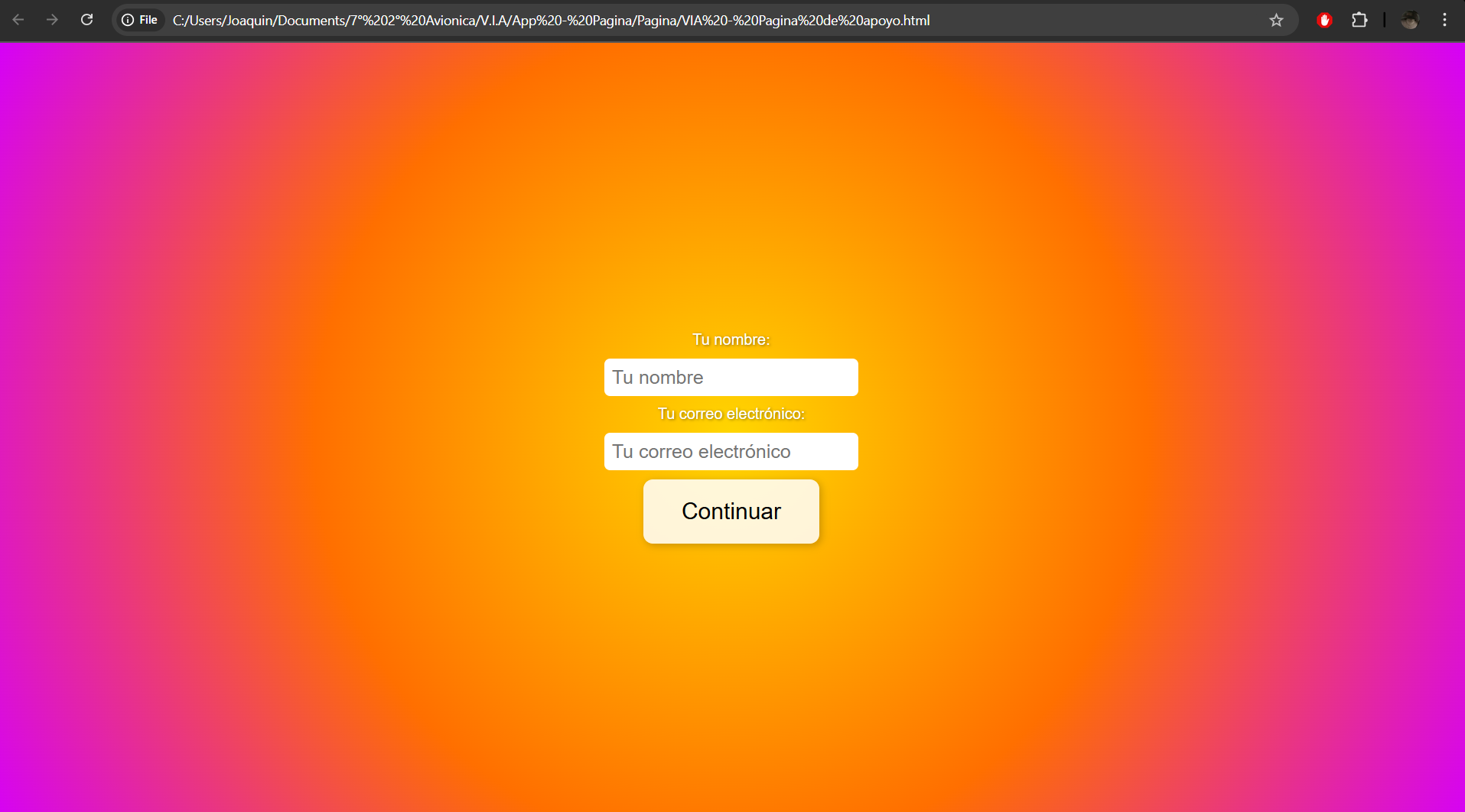
También, usando una página llamada ElevenLabs, creo un narrador de voz, el cual conecto a la página para que lea los botones por los que el dedo pasa. Tiene un par de errores aun (como que no lee correctamente cuando se mantiene por mucho tiempo el dedo), por lo que me dedico a arreglarlos.

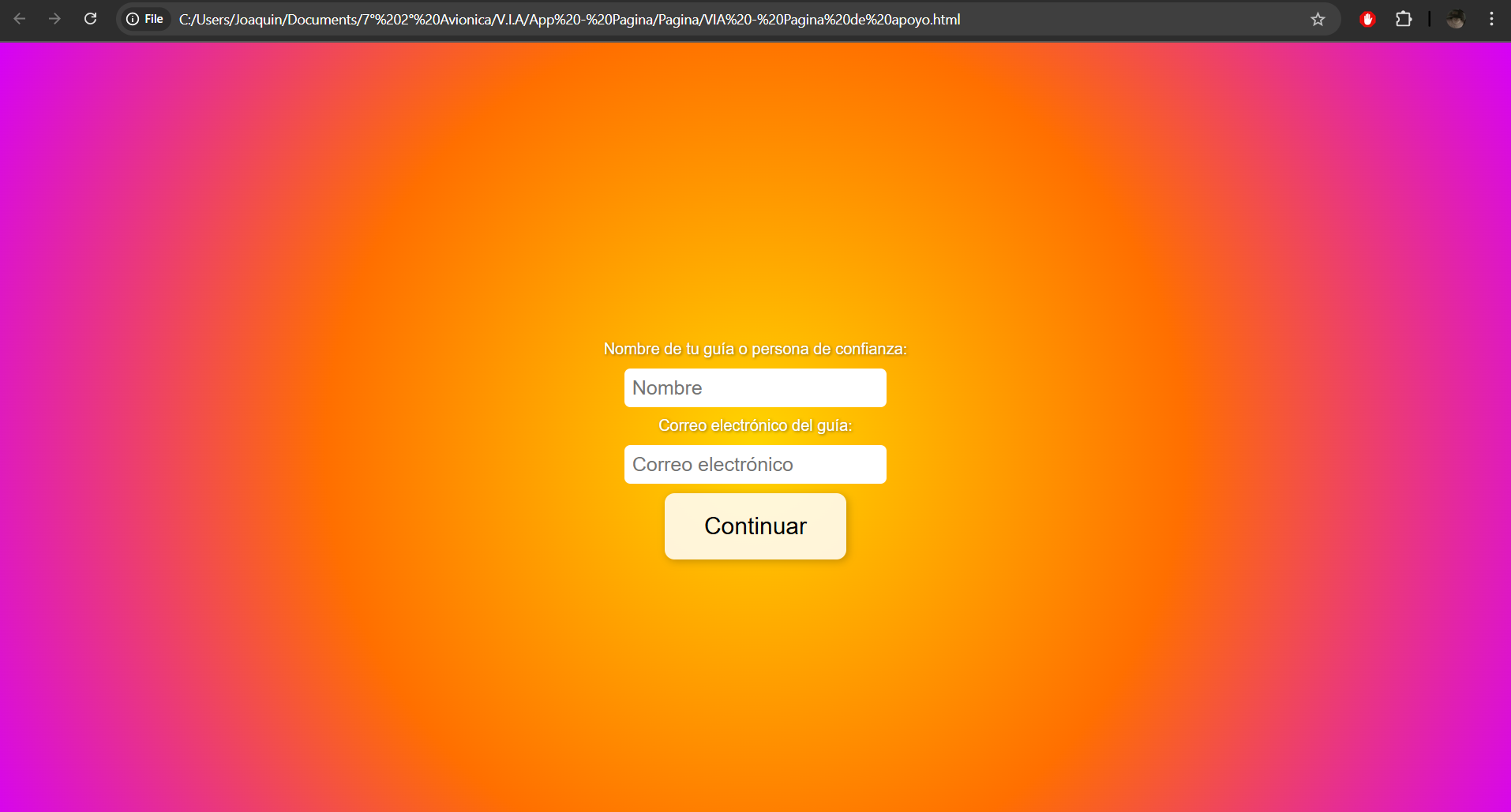
6/8 (miércoles 6 de agosto)

Ya con un transformador TTL a Serial, prueba una nueva versión del código, donde me permite sacar fotos y ver en el monitor si la imagen salió o no

12/8 (martes 12 de agosto)

Actualizo la sección de la información del usuario y de la persona de confianza, pidiendo el Gmail además del nombre





Aun así, este proceso aun no envía mails a las personas correspondientes. Así que, me dedico a investigar una manera de enviar mails de un mail a otro.

15/8 (viernes 15 de agosto)

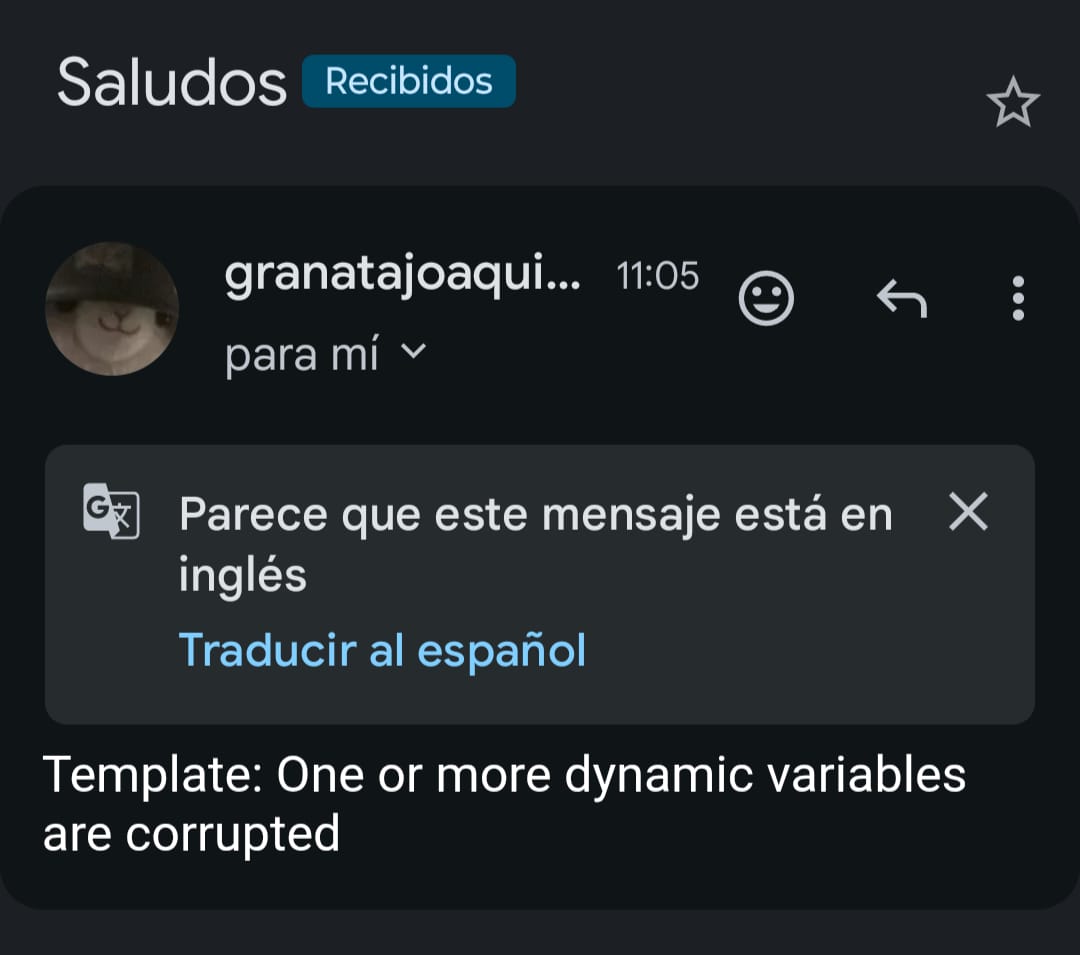
Después de un poco de investigación, encuentro esta página llamada EmailJS, que permite enviar un mensaje predefinido a un mail.

Usando esa página, tengo pensado enviar un mail de confirmación y otro cuando el usuario elija mandar la ubicación a la persona de confianza.

Por eso, me pongo a averiguar cómo funciona la página, como usarla en esta ocasión y que debo añadir a el código de la página para que funcione

19/8 (martes 19 de agosto)

Logro enviar un mail a la persona de confianza, pero me sale el mismo error siempre.



Me dedico a buscar la causa del problema y arreglarlo.

22/8 (viernes 22 de agosto)

Después de darle un par de vueltas, encuentro la solución a los audios y logro que suenen correctamente.

26/8 (martes 26 de agosto)

Hago la última actualización al código de prueba. Esta vez, me permitirá ver lo que ve la camara en tiempo real.

#include "esp\_camera.h"

#include "BluetoothSerial.h"

// --- Configuración Bluetooth ---

BluetoothSerial SerialBT;  // Puerto Bluetooth SPP

// --- Pines de la cámara (AI Thinker ESP32-CAM) ---

#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 setup() {

  Serial.begin(115200);

  Serial.println("Iniciando...");

  // --- Inicializar cámara ---

  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;

  config.frame\_size = FRAMESIZE\_VGA;

  config.jpeg\_quality = 12;

  config.fb\_count = 1;

  if (esp\_camera\_init(&config) != ESP\_OK) {

    Serial.println("Error al inicializar la cámara");

    while (true) delay(1000);

  }

  Serial.println("Cámara inicializada correctamente");

  // --- Iniciar Bluetooth ---

  if (!SerialBT.begin("ESP32CAM\_BT")) {

    Serial.println("Error iniciando Bluetooth SPP");

    while (true) delay(1000);

  }

  Serial.println("Bluetooth listo. Nombre del dispositivo: ESP32CAM\_BT");

  Serial.println("Conéctate desde la Raspberry y abre /dev/rfcomm0");

}

void loop() {

  // Si llega un comando Bluetooth:

  if (SerialBT.available()) {

    String cmd = SerialBT.readStringUntil('\n');

    cmd.trim();

    if (cmd.equalsIgnoreCase("IMG")) {

      // Capturar imagen y enviarla

      Serial.println("Solicitando imagen...");

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

      if (!fb) {

        SerialBT.println("ERR\_CAM");

        return;

      }

      // Enviar tamaño (4 bytes, big-endian)

      uint32\_t len = fb->len;

      uint8\_t header[4];

      header[0] = (len >> 24) & 0xFF;

      header[1] = (len >> 16) & 0xFF;

      header[2] = (len >> 8) & 0xFF;

      header[3] = len & 0xFF;

      SerialBT.write(header, 4);

      // Enviar imagen

      SerialBT.write(fb->buf, fb->len);

      SerialBT.flush();

      esp\_camera\_fb\_return(fb);

      Serial.println("Imagen enviada");

    }

  }

}

10/9 (miércoles 10 de septiembre)

Pruebo el sensor de proximidad fuera de la simulación. Para la prueba, queremos poder leer la distancia en el monitor y vamos a usar LEDs para probar la respuesta dependiendo la distancia. El código que usaremos será parecido al usado en la simulación.

16/9 (martes 16 de septiembre)

Después de varias pruebas y errores, logramos hacer andar el monitor, dándonos la distancia que lee el sensor. Ahora falta la prueba con los LEDs.

17/9 (miércoles 17 de septiembre)

Realizamos la prueba de los LEDs usando una modificación del código anterior. La prueba consiste de tres LEDs: Rojo, amarillo y verde. Dependiendo de la distancia, se prendera el LED correspondiente.

Si la distancia es mayor a X, se prende el verde.

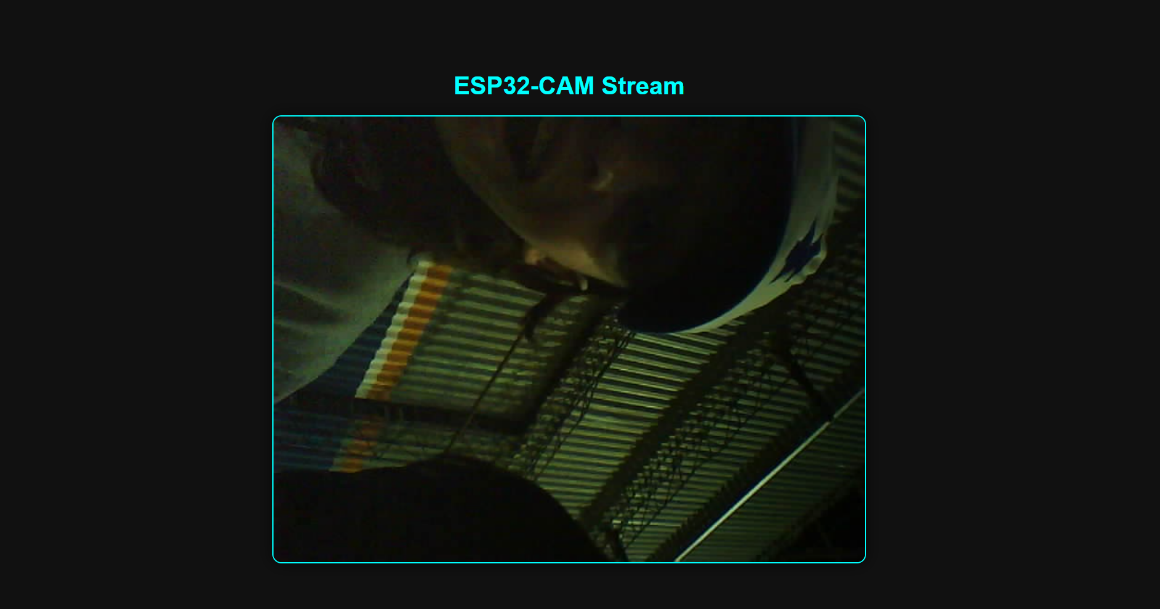
Si la distancia es menor a X, pero mayor a Y, se prende el amarillo.

Y si la distancia es menor a Y, pero mayor a 0, se prende el rojo.

23/9 (martes 23 de septiembre)  
Prosigo con la programación de la raspberry. Nuestra idea es enviar lo que la cámara recibe a la raspberry, donde esta identificara los distintos elementos, los copiara en un texto y los dictara por los auriculares. Para que esto funcione, debemos instalarle a la raspberry un programa que le permita identificar los elementos.

30/9 (martes 30 de septiembre)

Seguimos con la raspberry. Logramos que la cámara envié el video a la raspberry, además de permitirnos verlo desde el monitor conectado. Ahora, debemos añadir el sistema de detección de objetos



7/10 (martes 7 de octubre)

Probamos la detección de objetos. Para eso, usamos un modelo pre-entrenado, el cual ya tiene cargado varios elementos que puede detectar. El nombre de los objetos está en inglés y varios no eran necesarios, por lo que hay que moldear un poco el modelo.

8/10 (miércoles 8 de octubre)

Me dedico a probar, cambiar a español y ajustar el modelo. A su vez, hago que la raspberry genere texto de lo que detecta, para después usar un parlante o auricular para dictarlo.

14/10 (martes 14 de octubre)

Usando RealVNC y un monitor, me conecto a la raspberry y la conecto por bluetooth a un parlante para que después dicte lo detectado por la raspberry.

15/10 (miércoles 15 de octubre)

Después de probar el parlante, nos dimos de cuenta de algo gracioso. El parlante y la voz están en español, pero las palabras están escritas en inglés. Por lo que el parlante dirá las palabras en ingles con acento en español. En vez de Cell Phone, dirá Sel Pone

28/10 (martes 28 de octubre)

Primera clase de dron. Vemos las bases de volar drones, además de los peligros y conocimientos básicos de los mismos.

29/10 (miércoles 29 de octubre)

Armo una placa simple que mantenga de manera ordenada todos los componentes. Además, sirve para tener de manera cómoda un divisor de tensión que usaremos entre el sensor de proximidad y la ESP32.

4/5 (martes 4 de noviembre)

Segunda clase de dron. Vemos las cosas a tener en cuenta cuando vuelas un dron.

5/5 (miércoles 5 de noviembre)

Uso un cable grueso multifibra para conectar los elementos de la placa a la raspberry. Como el cable tenia pines macho de ambos lados, tuve que soldar cabezas de pines hembra a cada lado.