Nombre del proyecto: SmartSpace

Descripción: SmartSpace está enfocado en optimizar el consumo de energía mediante sensores y aprovechando las tecnologías ya existentes como cámaras de vigilancia.

Versión: 1

Fecha de creación: marzo del 2025

Autores:

Daniela Yvonne Zaragoza Vera.

Ana Laura Toledo Zepahua.

Rosa Maria Andrade Taboada.

Fernanda Daniela Pérez Álvarez.

INTRODUCCIÓN.

Problemática: En muchos entornos de trabajo como oficinas y centros de datos los sistemas de iluminación y climatización no están optimizados para las condiciones ambientales del espacio como consecuencia está el consumo innecesario de energía, que afecta a los costos operativos de la empresa, así como el bienestar de los empleados. Las variables como la luz natural que entra en el espacio, así como la cantidad de personas que se encuentra en el mismo son determinantes que provocan un gasto innecesario. A pesar de los avances tecnológicos las empresas siguen usando métodos o sistemas tradicionales que no responden de la mejor manera a las condiciones del entorno.

Objetivo: El principal objetivo del proyecto es optimizar el uso de recursos energéticos en los espacios de trabajo empresariales, monitoreando la luz y temperatura por medio de sensores además aprovecharemos tecnología de inteligencia artificial que puede trabajar en conjunto con cámaras que ya estén instaladas para el reconocimiento de objetos en este caso personas, así determinar la automatización de la climatización en el espacio.

Propósitos:

- Optimización de recursos energéticos: que regule el uso de la energía en los espacios de trabajo, adaptándose a las condiciones ambientales en tiempo real, para reducir el consumo innecesario de energía en iluminación y climatización.
- Monitoreo constante de variables ambientales: Implementar sensores de luz y temperatura que proporcionen datos precisos sobre las condiciones ambientales del espacio, permitiendo una gestión eficiente de los recursos energéticos en función de las necesidades del entorno.
- Integración de inteligencia artificial: Aprovechar las capacidades de la inteligencia artificial para procesar los datos provenientes de sensores y cámaras existentes en los espacios de trabajo, con el fin de identificar automáticamente la presencia de personas y ajustar los sistemas de climatización en consecuencia.
- Mejorar el bienestar de los empleados: Asegurar que las condiciones de trabajo sean óptimas en términos de temperatura y luz natural, promoviendo un ambiente más

cómodo y saludable para los empleados, lo que podría mejorar su productividad y satisfacción laboral.

- Reducción de costos operativos: Disminuir los costos asociados al consumo energético, al optimizar la iluminación y climatización según las condiciones reales del entorno, lo que resultará en un ahorro significativo para la empresa.
- Adaptación a tecnologías modernas: Reemplazar los sistemas tradicionales de control de energía por soluciones más inteligentes, adaptadas a las necesidades reales de los espacios de trabajo, con el fin de aprovechar las tecnologías disponibles para mejorar la eficiencia energética.

DESARROLLO.

¿Cuál es la idea de SmartSpace?

La idea es crear dos modelos que representen los sensores de luz y temperatura para tener una idea más clara de cómo será su funcionamiento, estos sensores estarán programados en Arduino para que se puedan detectar las métricas y valores necesarios, una vez que el código de Arduino ya pueda leer los valores de los modelos se debe conectar con la base de datos, la base de datos será desarrollada en Firebase pues así podrán hacerse las actualizaciones en tiempo real y ser agregadas de forma automática a la interfaz de usuario para que pueda visualizar los datos, la interfaz de usuario mostrara principalmente graficas con valores extraídos de Firebase estos datos como se menciona anteriormente serán actualizados en tiempo real y se puede analizar y navegar en la página web colocando el cursor sobre las gráficas para interactuar, como parte importante del proyecto los sensores en conjunto con el código trabajan para poder detectar una temperatura alta y de forma automática encender la calefacción para representar esta parte en el modelo se usara un motor que trabajara cuando el sensor detecte la temperatura limite, finalmente como extra de este proyecto se encuentra la parte de identificación de objetos a futuro este proyecto está pensado para que las cámaras que se encuentran en una empresa o establecimiento sean aprovechadas para monitorear la cantidad de personas que hay en un espacio, de acuerdo a esta variable se pueda regular la temperatura para disminuir o aumentar y que las condiciones ambientales sean adecuadas

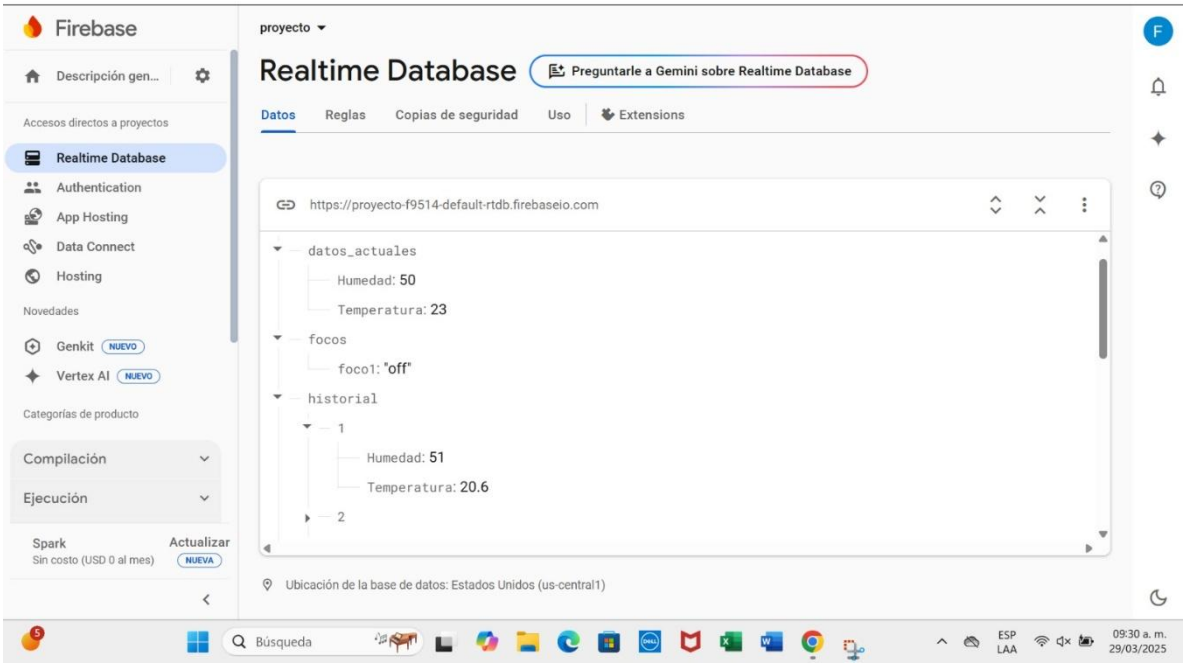
para los trabajadores, esta idea no solo es un beneficio para el personal también beneficia a la empresa ya que disminuye costos en el ahorro de energía.

Funcionalidad del código.

- Optimización de energía por medio de la regulación automatizada asegurando que no se consumirá energía innecesaria en espacios iluminados o con condiciones ambientales adecuadas.
- Detección de luz natural: el modelo utiliza un sensor fotorresistor que detecta la cantidad de luz natural y según la cantidad captada se modificara la intensidad de la luz.
- Encendido y apagado automático de luces: retomando el punto anterior el sistema es capaz de apagar por completo la luz en caso de que así se necesite, y de igual manera tener la luz al máximo.
- Ajuste de la intensidad de luces: mantiene el nivel adecuado para el entorno.
- Lectura de temperatura: por medio de sensores de temperatura se monitorean las condiciones del espacio de acuerdo con las necesidades.
- Detección de personas en espacios: por medio de una inteligencia artificial previamente entrenada, el sistema puede determinar la cantidad de personas en una sala de esta manera puede bajar o subir la temperatura de acuerdo con las aglomeraciones.
- Generación de gráficos: los datos recuperados de los sensores se grafican para proporcionar una visualización clara de los valores.
- Visualización de datos en tiempo real: los datos son visualizados en tiempo real lo que facilita la toma de decisiones.
- Aprovechamiento del uso de cámaras existentes: las cámaras que ya se encuentren instaladas serán aprovechadas para detectar personas en un espacio dejando a un lado la necesidad de instalar sensores adicionales.
- Interfaz gráfica de usuario: la interfaz con la que interactúa el usuario tendrá distintos apartados entre ellos las gráficas por área, y un apartado de alertas que nos mantendrá informados de las fallas.

Tecnologías para implementar en el proyecto.

- Arduino: útil para la automatización y control, su capacidad para interactuar con una amplia variedad de sensores lo hace para monitoreo de luz, temperatura y control de climatización. La razón para utilizar Arduino es su bajo costo y su facilidad de integración con hardware, bajo consumo de energía y la gran cantidad de bibliotecas y permite la implementación de IoT.
- Firebase: Firebase es la mejor opción para el proyecto debido a la automatización que queremos lograr, esta base actualiza en tiempo real los datos sin necesidad de recargar la página, eso quiere decir que se actualizan instantáneamente en la interfaz gráfica.



- Visual Studio Code: se usó este editor de código para crear la página web ya que es altamente flexible y personalizado debido al manejo de diferentes lenguajes se puede implementar css, HTML y JavaScript en un mismo programa y adema es compatible con Git lo que mejora el trabajo en equipo para una mayor eficiencia.

  De esta manera logramos una interfaz simple, entendible y atractiva para que el usuario pueda interpretar los valores graficados.

- YOLO: decidimos utilizar este software que está enfocado a IA ya que cuenta con entreno previo para lectura de objetos en tiempo real, esto nos ayudara a lograr la detección de personas en ciertos espacios y así poder regular la temperatura.

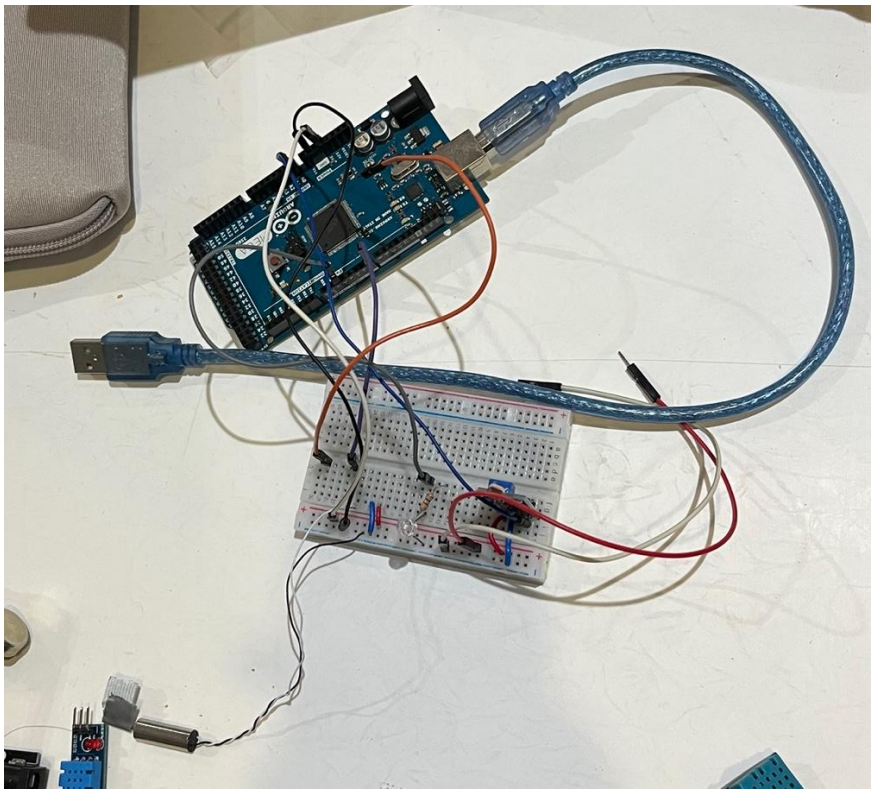
Material electrónico.

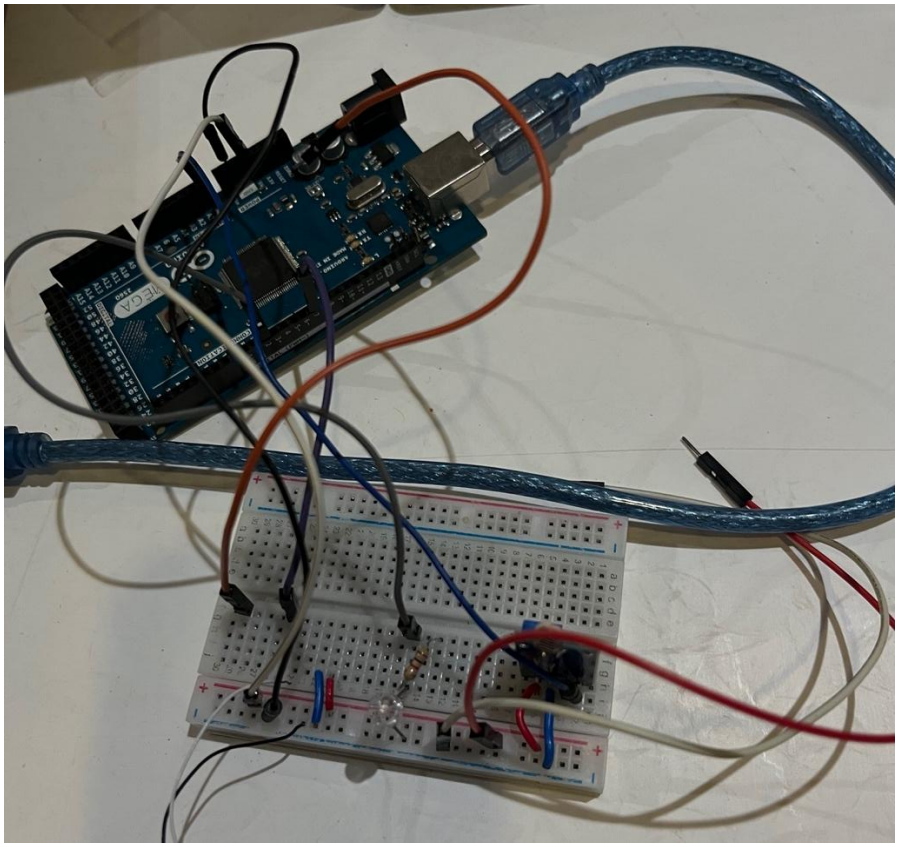
- 2 Protoboards.
- Jumpers.
- Led.
- Led de humedad y temperatura DH11.
- Modulo sensor fotoresistencia LDR.
- Resistencia.
- Arduino mega.
- Esp wroom 32.
- Motor.

El material anterior se utiliza para elaborar 2 modelos que trabajaran para simular los sensores del proyecto en la vida real, el primer modelo contiene el sensor fotorresistencia, led y resistencia se basa en una protoboard conectada por medio de jumpers al Arduino mega este modelo ya está programado para que el led se apague y prenda al detectar luz. Por otro lado, el segundo modelo trabaja con otro protoboard y led DH11 conectados entre sí por medio de jumpers.

Modelo 1.

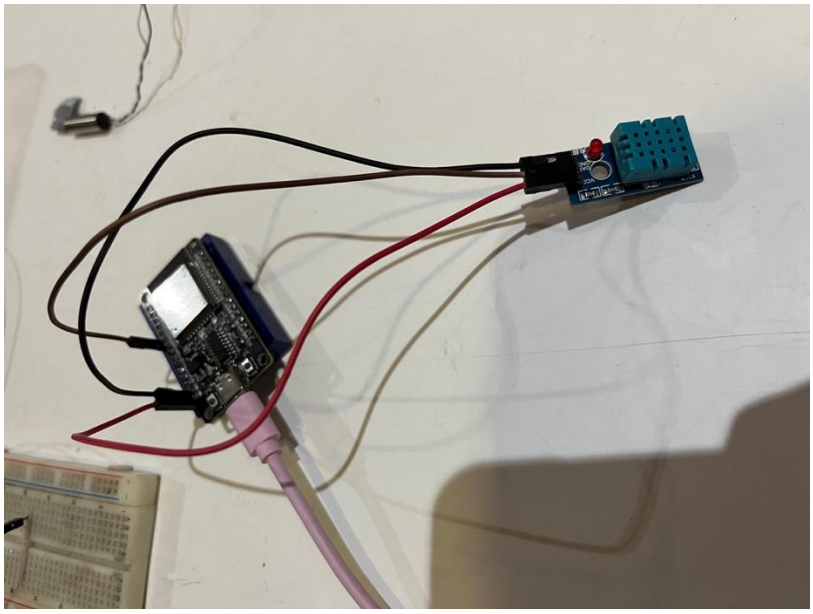
El sensor LDR es un tipo de resistencia cuyo valor cambia en función de la cantidad de luz que recibe. Es un componente electrónico que se utiliza como sensor de luz. Su resistencia disminuye cuando la luz impacta sobre él aumenta, y aumenta cuando la luz disminuye. Este cambio en la resistencia se puede medir y utilizar para controlar otros componentes, como encender o apagar un LED, en función de la cantidad de luz que detecte. Los LDRs son comúnmente utilizados en proyectos de detección de luz, como en sistemas de iluminación automática o en dispositivos de ahorro energético.

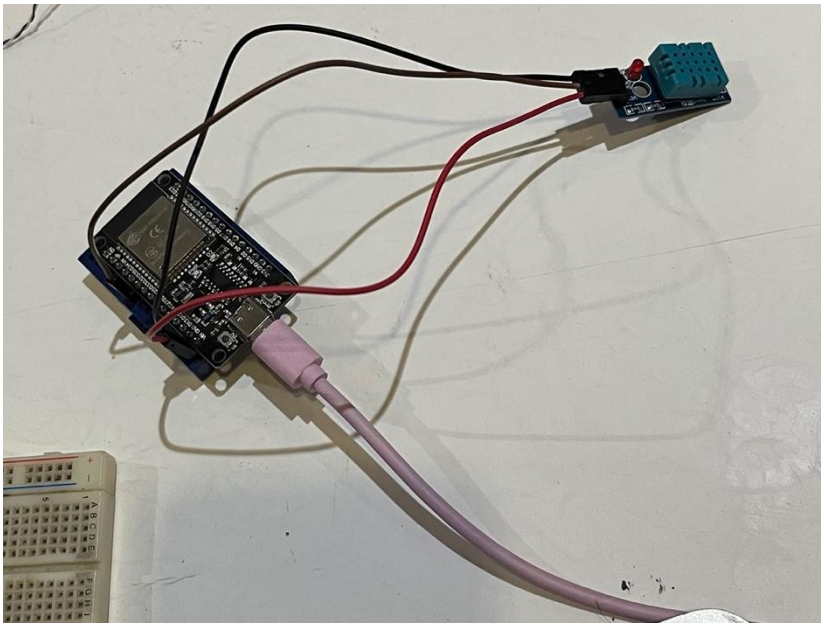




Modelo 2.

El segundo modelo utiliza un protoboard con un sensor DH11 conectado a través de jumpers al protoboard, que procesa los datos proporcionados por el sensor. Según los valores de temperatura y humedad detectados. Este modelo simula un sistema de monitoreo ambiental, donde el codigo vinculado puede mostrar los valores de temperatura y humedad o activar, como enviar una alarma, si los niveles alcanzan un valor crítico, permitiendo así controlar el ambiente en función de las mediciones obtenidas.





Código Arduino Modelo 1 – Sensor LED.

El código se encuentra en el repositorio de GitHub con el nombre “sensor_de_luz”.

A continuación, explicación de cada parte del código.

- Se definen los pines y las variables necesarias para que el sistema funcione correctamente en primer lugar se asigna el pin digital 32 al sensor DHT11, que se utilizará para medir la temperatura y la humedad. Luego, se define el tipo de sensor con la instrucción DHTTYPE DHT11, lo que indica que el sensor utilizado es un DHT11. A continuación, se asigna el pin analógico A0 al LDR (Light Dependent Resistor), el cual detecta la cantidad de luz ambiental.
- Dentro de la función setup(), que se ejecuta una sola vez al inicio del programa, se realiza la configuración inicial de los componentes del sistema. Primero, se establece la comunicación serie mediante Serial.begin(115200), lo que permite que el microcontrolador envíe datos al monitor serie a una velocidad de 115200 baudios. Esto es esencial para poder observar los valores de los sensores y las acciones del programa en tiempo real. Luego, el pin 2, al que está conectado el LED, se configura como una salida con la instrucción pinMode(ledPin, OUTPUT), permitiendo que el microcontrolador controle su encendido y apagado.
- En la función loop(), que se ejecuta continuamente, se realiza la lectura de los valores de los sensores de forma repetitiva. Primero, se lee el valor de luz del LDR utilizando la función analogRead(ldrPin), que devuelve un valor entre 0 y 1023, dependiendo de la cantidad de luz que incide sobre el sensor.
- En este paso, el programa utiliza el valor leído del LDR para controlar el encendido o apagado del LED. Si el valor del LDR es superior a 300, lo que indica que hay suficiente luz en el ambiente, el LED se enciende usando la función digitalWrite(ledPin, HIGH), y se imprime en el monitor serie que el LED está "encendido". Si el valor del LDR es menor o igual a 300, lo que indica que hay poca o ninguna luz, el LED se apaga utilizando digitalWrite(ledPin, LOW), y se muestra en el monitor serie que el LED está "apagado". Este control permite que el sistema responda automáticamente a los cambios de luz en el entorno, encendiendo el LED cuando sea necesario.

- El código imprime los valores de temperatura y humedad leídos del sensor DHT11 en el monitor serie. Esto se hace mediante las funciones Serial.print() y Serial.println(). Primero, se imprime la etiqueta "Temperatura: ", seguida del valor de temperatura almacenado en la variable temperature, a continuación, se imprime la etiqueta "Humedad: ", seguida del valor de humedad almacenado en la variable humidity, y se añade el símbolo de porcentaje.
- En este paso, el sistema toma decisiones sobre el ventilador en función de la temperatura medida por el sensor DHT11. Si la temperatura es mayor a 25°C, el sistema considera que el ambiente está demasiado caliente y enciende el ventilador utilizando la función digitalWrite(fan, HIGH), lo que hace que el ventilador empiece a funcionar.
- Por último, se introduce un pequeño retraso de 500 milisegundos con la función delay(500). Este retraso es útil para evitar lecturas excesivamente rápidas de los sensores, lo que podría sobrecargar al sistema y dificultar su funcionamiento estable. Además, proporciona tiempo para que el sistema se estabilice entre cada ciclo de medición.

Código Arduino Modelo 2 – Sensor de temperatura.

El código final se encuentra en el repositorio de GitHub con el nombre “datos_temperatura_y_humedad” este código está diseñado para usar un ESP32 con capacidades wi-fi y leer datos de un sensor DHT11 y enviar esos datos a una base de datos en tiempo real utilizando Firebase.

- Se incluyen las librerías necesarias para el funcionamiento del código. WiFi.h se usa para manejar la conexión Wi-Fi con la red, FirebaseESP32.h permite interactuar con la base de datos Firebase, y DHT.h se emplea para controlar el sensor de temperatura y humedad DHT11. Posteriormente, se definen las credenciales de la red Wi-Fi que se utilizarán para conectar el ESP32 a Internet.
- Luego, el código configura el sensor DHT11. El pin al que está conectado el sensor se define como 32, y el tipo de sensor se especifica como DHT11. Se crea un objeto dht11 que se usará para interactuar con el sensor a través de los métodos proporcionados por la biblioteca DHT.h.

- En la función setup(), el ESP32 comienza el proceso de conexión a la red Wi-Fi utilizando las credenciales proporcionadas. Durante este proceso, se imprime el mensaje "Conectando a WiFi" en el monitor serie y, cuando la conexión es exitosa, se muestra el mensaje "WiFi conectado". Si no se logra conectar, el programa espera y sigue intentando hasta que se conecte correctamente.
- Una vez conectado a la red Wi-Fi, el siguiente paso es establecer la conexión con Firebase. El código se conecta a la base de datos de Firebase utilizando las credenciales de FIREBASE_HOST (que contiene la URL de la base de datos) y FIREBASE_AUTH (la clave de autenticación necesaria). Además, se habilita la opción de reconectar automáticamente a Wi-Fi si la conexión con Firebase se pierde en algún momento.
- Dentro de la función loop(), el código lee constantemente los valores de temperatura y humedad del sensor DHT11 mediante los métodos readTemperature() y readHumidity(), respectivamente.
- Una vez que se leen los valores, el código procede a enviar estos datos a la base de datos Firebase utilizando Firebase.pushFloat(). Esta función permite enviar los valores de temperatura y humedad como datos flotantes a las rutas dentro de la base de datos. Si los datos se envían correctamente, se muestra un mensaje en el monitor serie indicando que la temperatura y la humedad se enviaron a Firebase. Si ocurre algún error, se imprime el motivo de la falla en el envío.
- Finalmente, el código introduce un delay(5000), lo que provoca una espera de 5 segundos antes de realizar una nueva lectura de los datos del sensor y enviarlos nuevamente a Firebase. Este ciclo se repite continuamente, permitiendo que los datos de temperatura y humedad se actualicen en tiempo real cada 5 segundos.

Código de la página web.

El código de la página web es muy extenso debido a que tiene estilos y diferentes partes como HTML, CSS y JavaScript en este caso la explicación que se muestra a continuación será de manera general con el fin de tener un poco más de contexto del código.

Claro, aquí tienes un resumen del código:

Este código genera tres gráficos de área usando ApexCharts, cada gráfico representa dos series de datos uno de temperatura y otro de humedad con sus respectivas variaciones a lo largo de un periodo de tiempo. La información se presenta en un formato visual que permite observar cómo cambian ambos parámetros a lo largo del tiempo.

- Los datos de entrada son temperatura y humedad que en el código son dos arrays que contienen valores de temperatura en grados Celsius y la humedad en porcentaje, para diferentes momentos del día, utiliza timestamps que es un array de cadenas con las marcas de tiempo en formato ISO indicando el momento especifico en toma de mediciones.
- La configuración del gráfico se da por medio de series que están clasificadas en temperatura y humedad el tipo de grafico es de área lo que significa que muestra áreas rellenas debajo de las líneas que representan cada serie de datos, el eje x está configurado para mostrar las marcas de tiempo, y el tooltip está configurado par amostrar la fecha y la hora en el formato adecuado cuando el usuario pasa el cursor sobre el punto.
- Los tres gráficos creados se encargan de renderizar cada gráfico en un contenedor específico dentro de HTML.
- Finalmente, el objetivo de este proyecto, como se puede ver en el repositorio y en esta breve información, es crear visualizaciones interactivas y autónomas de datos de temperatura y humedad para dar el seguimiento de como estos parámetros cambian a lo largo del tiempo.

Código YOLO.

El objetivo de este código es identificar personas a través de tiempo real por medio de una cámara web utilizando la herramienta YOLO, el código se encuentra en el repositorio de GitHub con el nombre “detector_de_personas” se resume en los siguientes pasos.

- Al inicio del código, se importan las librerías necesarias. cv2, que es la biblioteca de OpenCV, se usa para capturar y procesar las imágenes provenientes de la cámara web, mientras que YOLO se importa desde el paquete ultralytics para realizar la detección de objetos en las imágenes.

- Se carga el modelo preentrenado de YOLOv8 a partir del archivo "yolov8n.pt". Este modelo está previamente entrenado para reconocer varios tipos de objetos, como personas, coches, animales, entre otros.
- La cámara web se inicializa mediante cv2.VideoCapture(0). Esta función abre el flujo de video de la cámara predeterminada. Si no se puede acceder a la cámara, el programa imprime un mensaje de error y termina la ejecución.
- Dentro de un bucle que se ejecuta de manera continua, se capturan los fotogramas de la cámara utilizando el método cap.read(). Si ocurre un problema y no se logra capturar el fotograma, el programa muestra un mensaje de error y abandona el bucle.
- Para cada fotograma, el modelo YOLO realiza la detección de objetos con el comando model.predict(). Este paso genera una lista de objetos detectados en el fotograma, y el parámetro conf=0.5 asegura que solo se consideren aquellos objetos cuya probabilidad de detección sea mayor o igual al 50%. El modelo devuelve información sobre la ubicación de cada objeto en el fotograma y la clase del objeto.
- El programa recorre las detecciones devueltas por el modelo YOLO y verifica si el objeto detectado es una persona. Esto se realiza comparando la clase del objeto con la palabra clave "person" en el modelo. Si el objeto es una persona, se incrementa un contador person_count, y se dibuja un rectángulo verde alrededor de la persona en el fotograma para resaltarla visualmente.
- El número total de personas detectadas en el fotograma actual se muestra en la esquina superior izquierda de la ventana de visualización. Esta información se actualiza en tiempo real a medida que el modelo detecta personas en cada fotograma.
- El fotograma, ahora con las anotaciones (rectángulos y texto), se muestra en una ventana llamada "Detección de Personas".

RESULTADOS.

Los resultados obtenidos a lo largo del desarrollo de este proyecto son las gráficas automatizadas que se muestran en la página web que se muestran en seguida, como podemos ver los datos de Firebase se muestran en las gráficas ahorrando el tiempo de análisis ya que se hace el registro automático, claro y preciso.

Link de Github: https://github.com/DanielaZaragoza/Hackathon2025