DISEÑO E IMPLEMENTACIÓN DE SISTEMA IOT PARA MEJORA DE LA CONDUCCIÓN.

DESIGN AND IMPLEMENTATION OF IOT SYSTEM TO IMPROVE DRIVING

TRABAJO FIN DE MÁSTER
CURSO 2022-2023

AUTOR
EDWIN ANDRÉS MENDOZA ZÚÑIGA

DIRECTOR
CARLOS GARCÍA SÁNCHEZ

MÁSTER EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

DISEÑO E IMPLEMENTACION DE SISTEMA IOT PARA MEJORA DE LA CONDUCCIÓN.

DESIGN AND IMPLEMENTATION OF IOT SYSTEM TO IMPROVE DRIVING

TRABAJO DE FIN DE MÁSTER EN INGENIERÍA INFORMÁTICA

AUTOR
EDWIN ANDRÉS MENDOZA ZÚÑIGA

DIRECTOR
CARLOS GARCÍA SÁNCHEZ

CONVOCATORIA: JULIO 2023
CALIFICACIÓN: 5

MÁSTER EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

10 JUNIO 2023

Dedicatoria

Dedico este trabajo principalmente a Dios, por darme la oportunidad de permitirme haber llegado hasta este momento de mi vida tan importante de mi formación profesional. A mi familia en especial a mi madre, por ser el pilar más importante y por demostrarte que con esfuerzo, dedicación se puede cumplir las metas que se propone. A mi padre, a pesar de ya no estar presente sé que es un momento orgullo para él.

Edwin Andrés Mendoza Zúñiga

Agradecimientos

Agradezco a Dios por darme el privilegio de la vida y continuar con un camino de logros y triunfos, que a pesar de tan duros que sean me acompaña para lograrlos.

A mi madre, que con su ejemplo me ha enseñado a no rendirme y luchar por las cosas que con el pasar del tiempo son necesarias y me ha demostrado que un fracaso es una fortaleza para continuar luchando por un mañana mejor.

A mis hermanas Verónica, Lizeth y mi sobrino Aaron por ser una gran familia y continuar siendo imperfectos para lograr la perfección.

Edwin Andrés Mendoza Zúñiga

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

DISEÑO E IMPLEMENTACIÓN DE SISTEMA IoT PARA MEJORA DE LA CONDUCCIÓN.

En el presente proyecto se describe una solución IoT low cost para DDD (Driver Drowsiness Detection) el cual, al ser implementada en una flota de vehículos permitirá controlar los niveles de somnolencia y tener una planificación de tiempos de descanso para cada uno de los conductores.

Su implementación consiste de cuatro partes:

- **Configuración del ESP32-CAM:** en el cual esta implementada la configuración para la conexión Wifi con el ESP32-CAM para que la placa pueda tener conexión a internet, adicional tenemos la comunicación entre el servidor-cliente mediante cliente-sockets.
- **Algoritmo DDD:** Implementar un algoritmo de detección de somnolencia utilizando técnicas de visión por computadora. Esto implica el procesamiento de las imágenes capturadas para identificar patrones asociados con la somnolencia, como el cierre de los ojos o bostezo. Consiste en buscar la última imagen sin procesar, mediante la utilización de la librería dlib que permite obtener un mallado facial mediante 68 puntos de referencia del rostro. Además, se utiliza la librería face-recognition que se encarga del reconocimiento facial para identificar a la persona(driver).
- **Configuración BBDD:** su implementación consiste en recibir los datos tanto del servidor socket como enviar los datos a la WEB. Su propósito principal es proporcionar un medio para almacenar y recuperar información de manera rápida y confiable.
- **WEB:** su implementación consiste en la parte visual del análisis realizado por el Algoritmo DDD. Con ello se muestra los resultados que se encuentran cargados en la base de datos.

Palabras clave

IoT, dlib, fase-recognition, aprendizaje profundo, Arduino, mongodb, reconocimiento facial, esp32-cam. Bootstrap 5, php.

ABSTRACT

DESIGN AND IMPLEMENTATION OF IoT SYSTEM TO IMPROVE DRIVING.

This project describes a low-cost IoT solution for DDD (Driver Drowsiness Detection) which, when implemented in a fleet of vehicles, will make it possible to monitor drowsiness levels and plan rest times for each driver.

Its implementation consists of four parts:

- **Configuration of the ESP32-CAM:** in which the configuration for the Wifi connection with the ESP32-CAM is implemented so that the board can have an Internet connection, additionally we have the communication between the server-client through client-sockets.
- **DDD Algorithm:** Implement a drowsiness detection algorithm using computer vision techniques. This involves processing captured images to identify patterns associated with drowsiness, such as eye closure or yawning. It consists of searching for the last unprocessed image, using the dlib library to obtain a facial mesh using 68 facial landmarks. In addition, the face-recognition library is used, which is responsible for facial recognition to identify the person(driver).
- **Database configuration:** its implementation consists of receiving the data both from the socket server and sending the data to the WEB. Its primary purpose is to provide a means to store and retrieve information quickly and reliably.
- **WEB:** its implementation consists of the visual part of the analysis performed by the DDD Algorithm. This shows the results that are loaded in the database.

Keywords

IoT, dlib, phase-recognition, deep learning, Arduino, mongodb, facial recognition, esp32-cam. Bootstrap 5, php.

Capítulo 1 - Introducción

En la actualidad estamos viviendo en una era tecnológica en constante evolución. Los avances tecnológicos han tenido un impacto significativo en casi todos los aspectos de nuestras vidas, desde la forma en que nos comunicamos y trabajamos hasta cómo accedemos a la información y nos entretenemos.

La tecnología ha impulsado el desarrollo de áreas como la inteligencia artificial, el internet de las cosas, la robótica y la realidad aumentada, entre otras. Estas tecnologías emergentes están cambiando nuestra forma de interactuar con el mundo y abriendo nuevas posibilidades en campos como la medicina, la energía, el transporte y el medio ambiente.

Sin embargo, junto con los beneficios, también surgen desafíos y responsabilidades. Es importante abordar temas como la seguridad y la privacidad de los datos, la brecha digital y el impacto ambiental de la tecnología. La ética y la regulación juegan un papel fundamental en asegurar que la tecnología se utilice de manera responsable y en beneficio de la sociedad en su conjunto.

En resumen, estamos inmersos en una era tecnológica en la que la innovación y el cambio son constantes. La tecnología está transformando nuestra forma de vivir, trabajar y relacionarnos, y seguirá siendo un factor clave en el desarrollo futuro de la sociedad.

Por esta razón, el objetivo de este trabajo es diseñar un sistema de detección de somnolencia utilizando el ESP32-CAM que sea capaz de identificar signos de adormecimiento en una persona, como el cierre de los ojos o el bostezo, mediante una interfaz intuitiva y fácil de usar que permita a los usuarios una retroalimentación sobre su comportamiento al volante cuando experimentan cansancio o fatiga.

1.1 Motivación

Según el estudio realizado por el doctor Carlos Egea [1], miembro de la Sociedad Española de Sueño (SES), ha alertado de que la falta de sueño mientras se conduce es comparable a tener una tasa de alcoholemia en sangre de 0,10ml.

Desde el 1 de julio, todos los coches nuevos homologados en Europa (así como los nuevos que se vendan a partir de 2024) tendrán que llevar instalado el sistema ADAS (Advanced Driver Assistance Systems) [2], una evolución de los sistemas ABS y ESP que, entre otras funcionalidades, deberá incorporar un detector de sueño. Mediante una cámara que enfoca a la cara del conductor, identificará síntomas de sueño y cansancio y, a través de señales de alarma visuales y sonoras, le avisará de que necesita detener su vehículo en un lugar apropiado y tomarse un descanso [2].

Esta nueva herramienta pretende acabar con uno de los principales motivos que se esconden tras los accidentes de tráfico. "Se estima que entre un 20 y un 30 por ciento de los accidentes de tráfico pueden atribuirse a la somnolencia", afirma el doctor Egea [2].

Según datos de la Dirección General de Tráfico (DGT), solo en 2020 más de 300 personas perdieron la vida en las carreteras a causa de las distracciones al volante, entre las que el sueño y la fatiga son "factores de riesgo desconocidos pero que concurren en la accidentalidad vial". [2]

El cansancio en el lugar de trabajo puede afectar la productividad y aumentar el riesgo de accidentes laborales. Por ello he desarrollado este proyecto de sistema de detección de cansancio para alertar a los trabajadores y empleadores cuando es necesario tomar medidas para prevenir accidentes o mejorar el rendimiento laboral, de una manera sencilla y práctica.

1.2 Análisis de soluciones existentes

En esta sección se muestran algunas alternativas al sistema implementado en este trabajo.

1.2.1 Sistemas ADAS

Los sistemas avanzados de ayuda a la conducción (ADAS por sus siglas en inglés) son sistemas que permiten mejorar la seguridad del vehículo, tanto de cara a sus ocupantes como al resto de usuarios de la vía. Algunos de ellos son capaces de tomar el control del vehículo en determinadas circunstancias para evitar un siniestro o minimizar sus consecuencias [3].

Alertas que proporciona el sistema ADAS:

a. **Frenado autónomo de emergencia:** esta función tiene como primer objetivo el alertar al conductor cuando está cercano a un posible peligro (otro vehículo, un peatón, un objeto en la ruta, etc.) y por medio de una señal (sonora y/o parpadeante) indica que se debe reducir la velocidad del vehículo para mantener una distancia de seguridad. En el caso de que la velocidad no se reduzca se activa el frenado de emergencia [4].

b. **Alerta de tráfico cruzado:** esta alerta funciona cuando un vehículo está yendo en marcha atrás y el conductor tiene poca visibilidad de lo que ocurre en la parte trasera. Es decir, si se cruza un vehículo más pequeño, una persona o un animal, los sensores inteligentes emiten una señal de alerta. Esto es muy útil, sobre todo para vehículos de gran dimensión, donde la visibilidad trasera se reduce considerablemente [4].

c. **Detección de ángulo muerto:** si bien los espejos retrovisores de los vehículos se pueden ajustar para que el conductor pueda ver por los costados y hacia atrás, siempre existen "puntos ciegos" donde el conductor no tiene rango de visibilidad. Para evitar accidentes debido a esto, la detección de ángulo muerto funciona gracias a unos sensores ubicados en ambas partes lateral trasera, los que emiten señales en caso de proximidad peligrosa [4].

d. **Sistema de detección de fatiga:** ya sea por cansancio, calor, enfermedad o por conducir sin detenciones más de las horas establecidas, la detección de fatiga supervisa y analiza el comportamiento de los conductores, detectando cuando existe algo fuera de lo habitual. En el caso de que el sistema detecte algún comportamiento sospechoso de fatiga o somnolencia, enviará una alerta (sonora, visual y/o táctil) para captar la atención del conductor. De ser necesario se le indicará que detenga el vehículo y tome un descanso [4].

e. **Mantenimiento activo en el carril**: esta función se activa cuando se detecta que el vehículo va a cambiar de carril de manera involuntaria, es decir, con sospecha de un posible accidente. El sistema analiza la velocidad, los movimientos del volante, la luz intermitente, y en base a todos estos parámetros, analiza si es un cambio de carril involuntario, generando una alerta para evitar un accidente [4].

A continuación, se explica en más detalle el Sistema de detección de fatiga:

El sistema elaborado por **Fundación MAPFRE**[1], recoge un listado importante de sistemas tecnológicos que ayudarían a evitar gran parte de los accidentes de tráfico si éstos se incorporaran en los vehículos. Entre ellos se encuentra el sistema de detección de fatiga. [5]

Dicho sistema alerta al conductor si pierde la concentración al volante, ya sea por fatiga o sueño. De esta forma, el conductor puede detener el vehículo hasta que se encuentre en perfecto estado para seguir conduciendo. [5]

Una propuesta de detección de somnolencia se basa en el aprendizaje de la red neuronal artificial (ANN por sus siglas en inglés, Artificial Neural Network)[2] y la fusión

[1] Fundación Mapfre disponible en https://www.fundacionmapfre.org/ (último acceso: 21-06-2023)

[2] Red Neuronal Artificial disponible en https://gamco.es/glosario/red-neuronal-artificial-ann/ (último acceso: 21-06-2023)

de datos de varios indicadores optimizados obtenidos de las señales físicas y de rendimiento de conducción del conductor. La Figura 1-1 se muestra la arquitectura general del sistema.

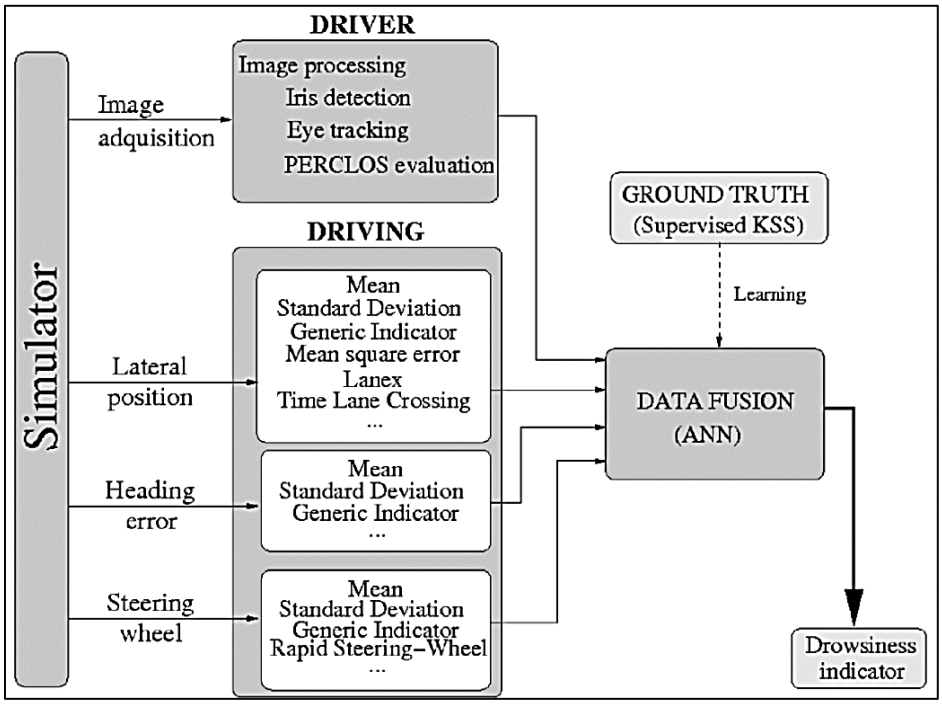

Figura 1-1 Arquitectura Sistema ADAS

1.2.2 Sistema de la compañía BOSH

Un sistema de la compañía BOSH **[6]** se centra en el algoritmo de la detección de somnolencia del conductor el cual monitorear los movimientos de la dirección y aconseja a los conductores que tomen un descanso a tiempo y así ayudar a prevenir accidentes causados por la fatiga y el micro sueño.

La detección de somnolencia del conductor es una función de software desarrollado por Bosch que analiza constantemente el comportamiento de dirección del conductor **[6]**.

Reconoce fases en que el conductor deja de conducir brevemente pero luego corrige bruscamente la dirección del vehículo, estas fases son un signo de disminución de la concentración y aumento fatiga.

El software reconoce señales de somnolencia, una señal audible, visual y/o háptica advierte al conductor que está cansado y recomienda un descanso. **[7]**

Figura 1-2 Modelo Bosh

1.3 Objetivos

1.3.1 Objetivo Principal

Según la investigación de soluciones existentes descritas en el apartado anterior, en este trabajo se propone la creación de un sistema IoT low-cost implementado en una flota de vehículos que permitirá controlar los niveles de somnolencia y tener una planificación de tiempos de descanso para cada uno de los conductores.

1.3.2 Objetivos Específicos

- **Objetivo 1:** Implementar un algoritmo de detección de la somnolencia utilizando tecnologías como el reconocimiento facial, monitoreo del parpadeo o la detección del bostezo.

- **Objetivo 2:** Integrar un sistema de alerta que notifique al usuario cuando se detecte somnolencia. Esto puede incluir la reproducción de un sonido, el envío de una notificación a un dispositivo móvil o la activación de una alarma.

- **Objetivo 3:** clasificar a los usuarios en grupos o segmentos con características y necesidades similares que permita la implementación del sistema en una flota de vehículos.

- **Objetivo 4:** Diseñar una interfaz de usuario para visualizar los resultados de la detección de somnolencia. La visualización en tiempo real de las imágenes capturadas, el registro de eventos de somnolencia y la presentación de estadísticas relacionada

1.4 Plan de trabajo

Para la elaboración de este proyecto se ha utilizado el sistema de Gatt, el cual consta de las siguientes partes:

La primera etapa consiste en realizar la descarga e instalación de Arduino IDE [3] , una vez realizado la instalación se procede a tener la información necesaria para el desarrollo del código. Realizadas las pruebas respectivas se procede a implementar en la placa ESP32-CAM y verificar el correcto funcionamiento del código.

La segunda etapa consiste en la investigación y familiarización con las tecnologías existentes relacionadas con la detección de la somnolencia, librerías como dlib, OpenCV y algoritmos de aprendizaje automático.

La tercera etapa consiste en el desarrollo del código para el servidor que escucha en una dirección IP y los puertos específicos. Cuando se recibe una conexión del cliente muestra la dirección IP. Después recibe los datos enviados por el cliente (ESP32-CAM) y los procesa.

La cuarta etapa consiste en la configuración de MongoDB[4] y el servidor de sockets que son componentes independientes. Para ello lo primero que debemos hacer es instalar MongoDB, instalar las bibliotecas necesarias y establecer la conexión con la base de datos en el servidor sockets.

[3] ARDUINO IDE disponible en https://www.arduino.cchttps://www.arduino.cc (último acceso: 21-06-2023)

[4] MongoDB disponible en https://www.mongodb.com (último acceso: 22-06-2023)

En la quinta etapa consiste en el desarrollo de una aplicación web que extraiga datos de la base de datos, para ello debemos configurar nuestro entorno de desarrollo web y un marco de trabajo para facilitar la creación del servidor web.

En la sexta etapa se expone la unión de toda la arquitectura del sistema, que incluye el ESP32-CAM, la base de datos MongoDB, un servidor para ejecutar el algoritmo de detección de somnolencia, el servidor sockets y una aplicación web.

1.5 Diagrama de Gantt

SISTEMA IOT
PARA MEJORA DE LA CONDUCCIÓN

Empresa: onRumboIoT
Responsable: Andrés Mendoza Z.

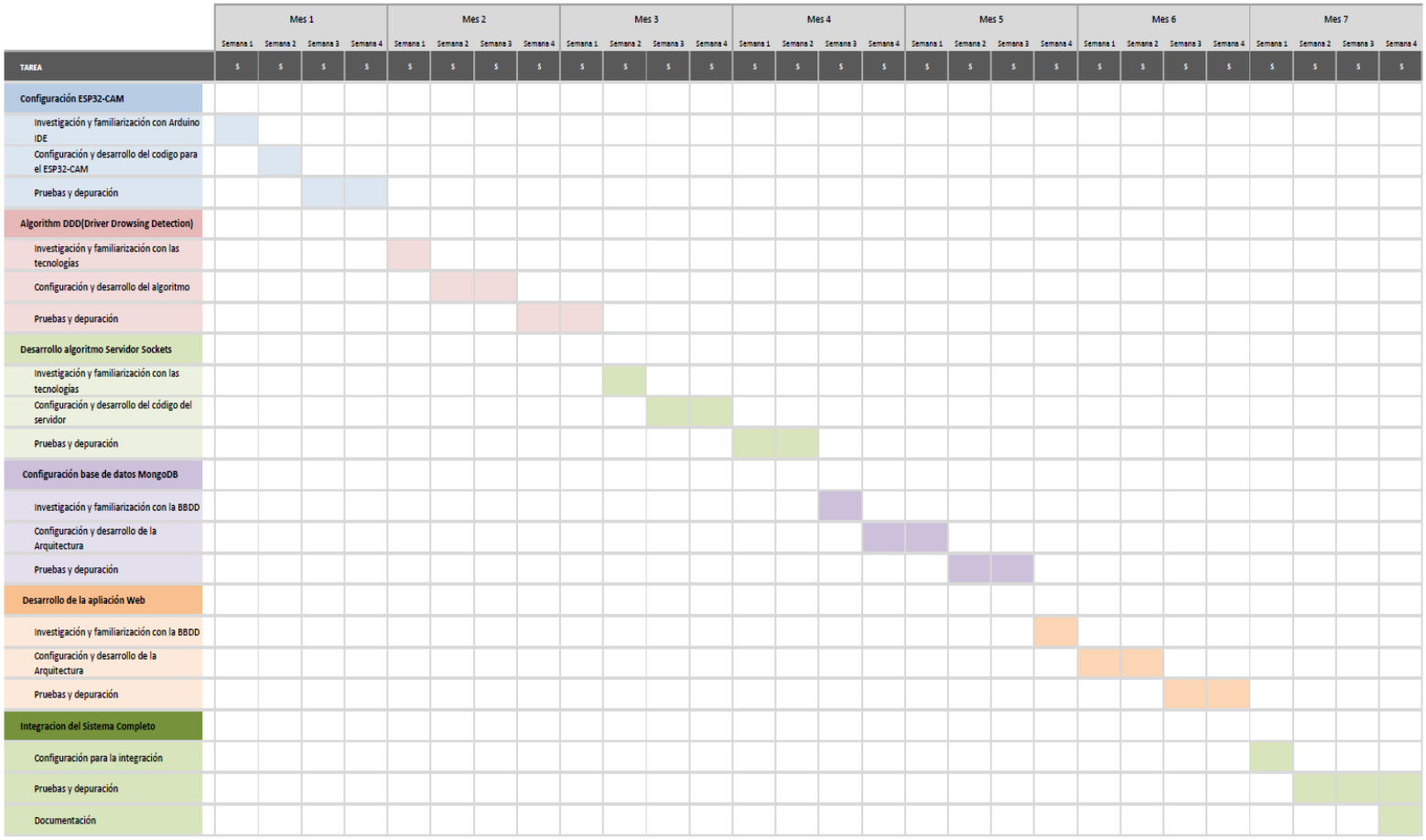

Figura 1-3 Diagrama de Gantt

1.6 Estructura del documento

Primer capítulo consiste en la introducción, describiendo la motivación, objetivo principal y los objetivos específicos a alcanzar en el proyecto, las soluciones existentes y el plan de trabajo que tomara el desarrollo del sistema.

Segundo capitulo consiste en una breve introducción al mundo del IoT, la propuesta del sistema IoT a desarrollar, diseño del sistema y la descripción de las partes que compondrán el sistema.

Tercer capítulo consiste en la investigación y familiarización con las tecnologías existentes relacionadas con la detección de la somnolencia, librerías como dlib, OpenCV y algoritmos de aprendizaje automático.

Cuarto capítulo nos centraremos en la creación de la arquitectura y la configuración de cada componente de nuestro proyecto. Esto nos ayudara a tener claro los protocolos de comunicación que nos van a permitir integrar las diferentes partes del sistema propuesto.

Quinto capítulo se expone la experimentación de os resultados obtenidos en el conjunto de pruebas, resultados del sistema realizando los debidos ajustes que demuestre el correcto funcionamiento del sistema implementado.

Sexto capítulo se expone las conclusiones del análisis de los resultados obtenidos durante el desarrollo de la aplicación, así como identificar áreas de mejora y posibles expansiones de la aplicación para trabajos futuros.

Por último, encontraremos las referencias bibliográficas utilizadas.

Capítulo 2 - Introducción al IoT, Propuesta, Diseño y descripción del sistema.

2.1 Introducción al IoT

En 1999 Kevin Ashton[5] inventa "Internet of things". El termino se refiere a la red colectiva de dispositivos conectados y la tecnología que facilita la comunicación entre dispositivos y la nube, así como entre los propios dispositivos [8].

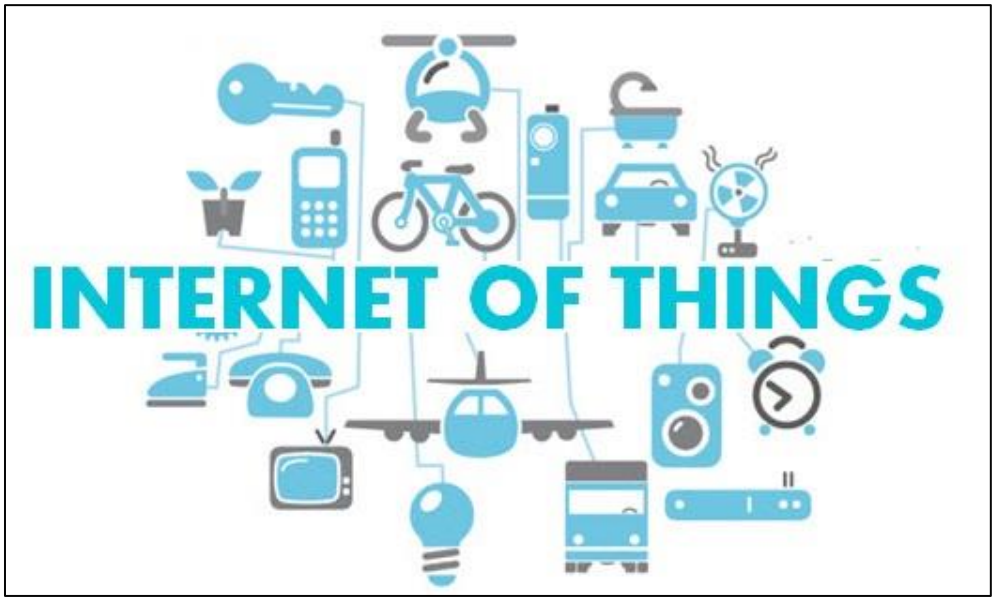

Figura 2-1 Internet of Things

2.1.1 ¿Como funciona el IoT?

Funciona mediante la recopilación de datos en tiempo real. El cual tiene tres componentes:

Dispositivo Inteligente: son los dispositivos como cámaras de seguridad, televisores, reloj inteligente los que tienen capacidades de computación para interconectarse con otros dispositivos [8].

[5] Kevin Ashton disponible en https://www.eexcellence.es/expertos/kevin-ashton-un-tecnologo-visionario (último acceso: 15-06-2023)

Aplicación Iot: es el conjunto de servicios y software que integra los datos recibidos del dispositivo [8].

Interfaz de usuario: Aplicación móvil o web que pueda utilizarse para registrar y controlar los dispositivos inteligentes. [8]

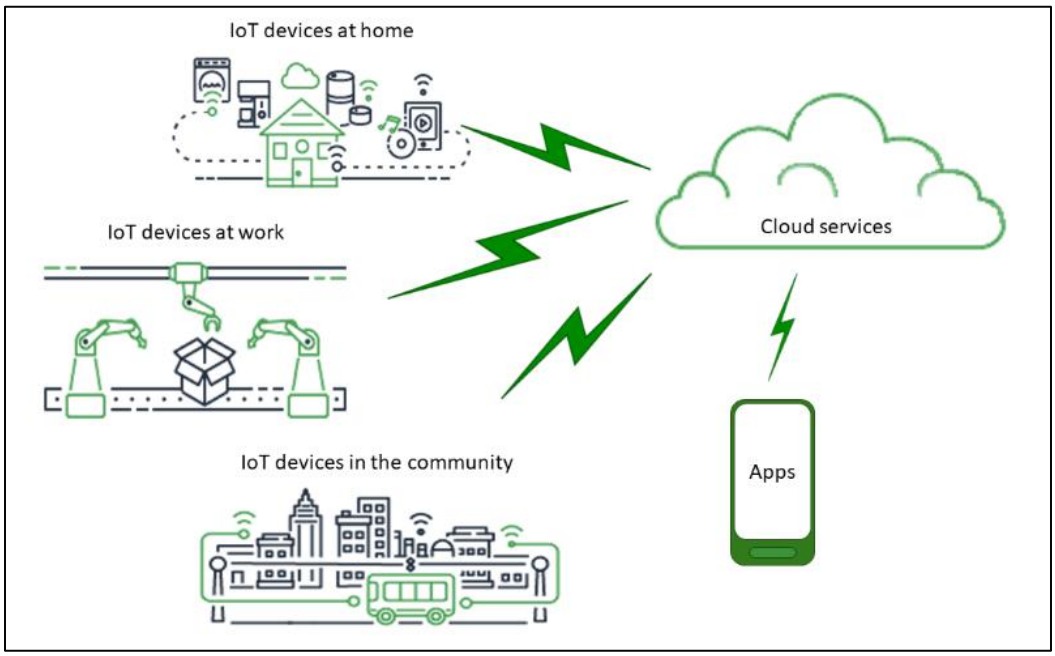

Figura 2-2 Funcionamiento IoT

2.2 Propuesta del sistema IoT

En este trabajo se propone la creación de un sistema IoT que ayude a los usuarios a medir y controlar los niveles de somnolencia en la conducción, mediante la integración de diferentes tecnologías como el ESP32-CAM que se encarga de capturar imágenes y enviarlas al servidor sockets, el mismo que envía las imágenes al servidor de inferencia para procesarlas y devolver al servidor de sockets. Los datos procesados son enviados a la base de datos de MongoDB para luego ser enviados a la web.

En la Figura 2-4 se observa el diseño del sistema propuesto.

2.3 Diseño del Sistema propuesto

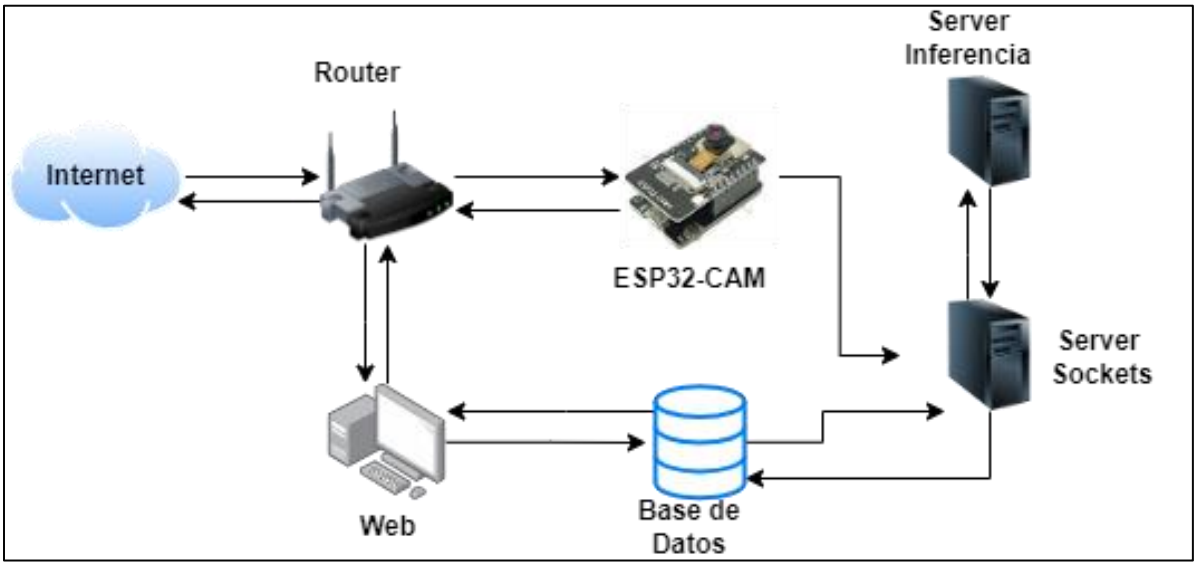

Figura 2-3 Diseño del Sistema propuesto

2.4 Descripción de sus partes

A continuación, se describe cada una de las partes que componen el sistema se somnolencia.

2.4.1 ESP32-CAM

Placa base ESP32-CAM[6]

ESP32-CAM, es un dispositivo que puede llamarse un todo en uno. Aparte de la conectividad Wifi y Bluetooth que viene de fábrica, pines GPIO, se le han añadido dos opciones más. Lleva integrado una pequeña cámara de video y una conexión para una tarjeta MicroSD, donde podremos almacenar fotos o videos [9].

[6] ESP32-CAM disponible en https://www.espressif.com/en/news/ESP32_CAM (último acceso: 22-06-2023)

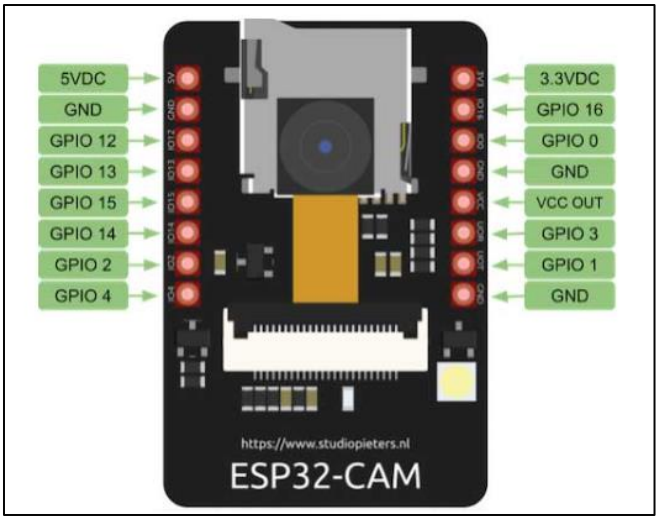

Figura 2-4 Esquema ESP32-CAM

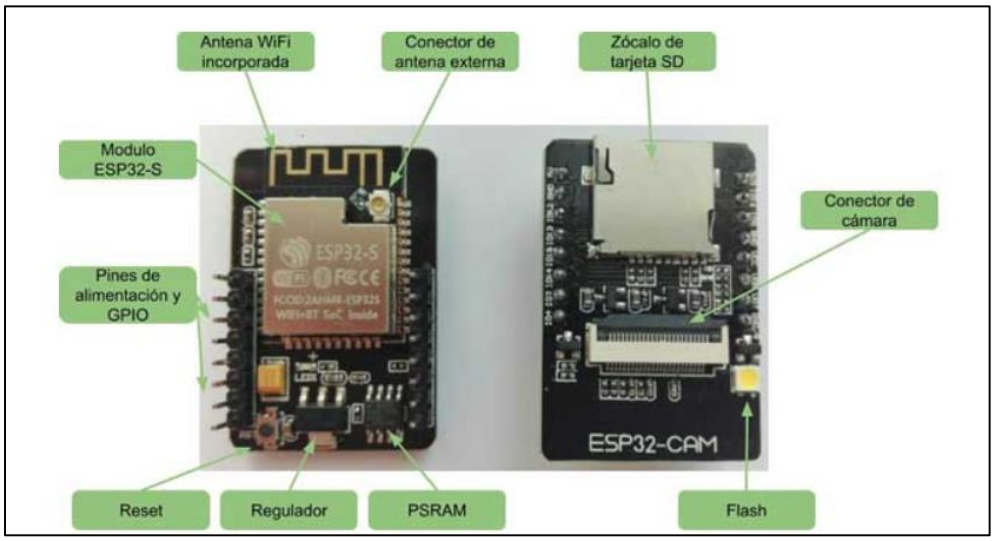

Figura 2-5 Partes importantes de ESP32-CAM

El ESP32-CAM no está diseñado específicamente para capturar imágenes en condiciones de oscuridad total, ya que no cuenta con un sensor de luz infrarroja o un iluminador integrado.

Sin embargo, emplear técnicas de procesamiento de imágenes después de capturar una imagen, como mejora de contraste, reducción de ruido o aumento de la exposición, puede mejorar la visibilidad en condiciones de poca luz.

ESP32-CAM es un módulo que combina un microcontrolador ESP32 y una cámara. Aunque el ESP32 tiene capacidad de conectividad Wi-Fi y Bluetooth, no cuenta con un hardware específico para admitir tarjetas SIM y la comunicación celular [10].

Al encontrarnos con la limitante de conexión del ESP32-CAM mediante un AP (Acces Point) se recomienda la integración de un módulo GSM para tener acceso a internet y poder transmitir los datos.

Para tener acceso a la red celular y utilizar una tarjeta SIM, necesitarías un módulo adicional que tenga soporte para tecnologías móviles, como GSM, GPRS o 3G/4G. Estos módulos, como el SIM800L o SIM900, se pueden conectar al ESP32-CAM a través de interfaces como UART (Universal Asynchronous Receiver/Transmitter) para habilitar la comunicación celular [10].

Al utilizar un módulo celular junto con el ESP32-CAM, puedes aprovechar la conectividad celular para enviar y recibir datos, como imágenes capturadas por la cámara, a través de la red celular. Sin embargo, ten en cuenta que la implementación de la comunicación celular requerirá conocimientos adicionales sobre el manejo de módulos celulares y la configuración de la red celular en tu país o región [10].

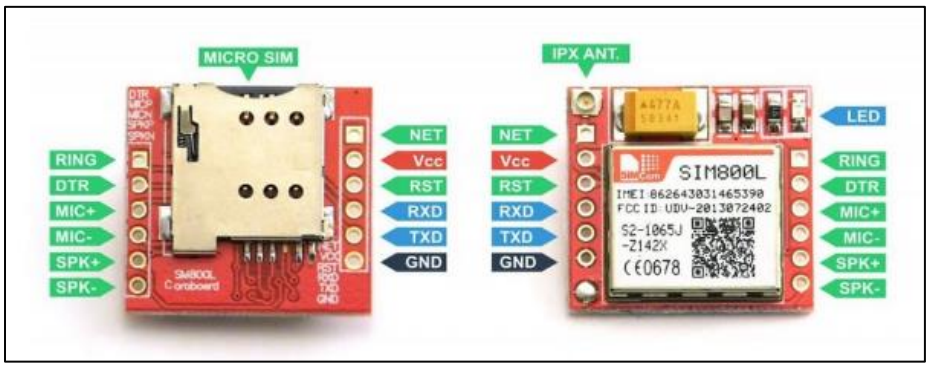

Figura 2-6 Características modulo SIM800L GSM

Al no ser posible implementar directamente una tarjeta SIM en un ESP32-CAM, se pudo verificar que el ESP32-CAM soporta módulo como el SIM800L[7] que se puede conectar al ESP32-CAM a través de interfaces como UART (Universal Asynchronous Receiver/Transmitter) para habilitar la comunicación celular, lo cual se podría implementar uno de los hardware.

2.4.2 MongoDB

Es un motor de la base de almacenamiento de datos NoSQL que se enfoca en documentos. Su estructura es muy parecida a la base de datos de JSON [11].

La función principal es de proporcionar un lugar para almacenar datos de manera estructurada y persistente. Para luego gestionar y recuperar los datos almacenados de una manera rápida y eficiente mediante consultas y búsquedas.

2.4.3 Algorithm DDD (Driver Drowsiness Detection)

Conocida como la detección de la somnolencia, se refiere al proceso de identificar y reconocer los signos y síntomas asociados a la fatiga y la falla de atención. El objetivo es detectar cuando una persona está presentando síntomas de cansancio muy elevados.

2.4.4 Servidor de Inferencia

Los servidores operan a través de una arquitectura llamada cliente-servidor[8]. Los servidores son programas de computadora en ejecución que atienden las peticiones de otros programas: los clientes[9]. Por tanto, el servidor realiza otras tareas para beneficio de

[7] SIM800L lo podemos encontrar en https://github.com/SETISAEDU/SIM800L-con-Arduino (último acceso: 22-06-2023)

[8] Cliente-servidor lo podemos encontrar en Cliente-servidor - Wikipedia, la enciclopedia libre (último acceso: 22-06-2023)

[9] Clientes lo podemos encontrar en Cliente (informática) - Wikipedia, la enciclopedia libre (último acceso: 22-06-2023)

los clientes; les ofrece la posibilidad de compartir datos, información y recursos de hardware[10] y software[11]. [12]

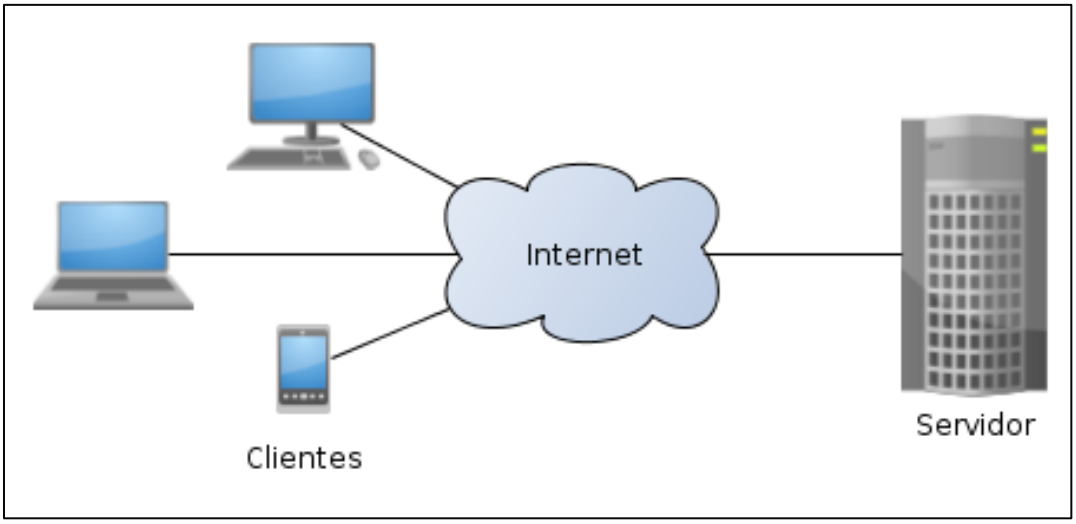

Figura 2-7 Cliente-Servidor

2.4.5 Aplicación Web

Una aplicación web es un software que se ejecuta en el navegador web. Su principal función es intercambiar información y proporcionar servicios de forma remota. Utilizan aplicaciones web para comunicarse con los clientes cuando lo necesiten y de una forma segura. [13].

[10] Hardware lo podemos encontrar en Hardware - Wikipedia, la enciclopedia libre (último acceso: 22-06-2023)

[11] Software lo podemos encontrar en Software - Wikipedia, la enciclopedia libre (último acceso: 22-06-2023)

Figura 2-8 Aplicación Web

Capítulo 3 - Tecnologías Utilizadas

Este capítulo se corresponde con la descripción de las distintas tecnologías existentes en la que se basa el sistema propuesto.

3.1 Arduino IDE[12]

El IDE es un conjunto de herramientas de software que permiten a los programadores desarrollar y grabar todo el código necesario para hacer que nuestro Arduino funcione como queramos. El IDE de Arduino nos permite escribir, depurar, editar y grabar nuestro programa (llamados "sketches" en el mundo Arduino) de una manera sumamente sencilla, en gran parte a esto se debe el éxito de Arduino, a su accesibilidad [14].

Arduino IDE permitió desarrollar el código para ser implementado en la placa de desarrollo ESP32-CAM. Lo cual su configuración permitió la captura y envió de imágenes correctamente.

3.2 OpenCV

Esta biblioteca se utiliza a menudo para el procesamiento de imágenes y vídeo. El módulo CV2 es el más importante de OpenCV, ya que proporciona a los desarrolladores una interfaz fácil de usar para trabajar con funciones de procesamiento de imágenes y vídeo [15].

La librería OpenCV proporciona un marco de trabajo de alto nivel para el desarrollo de aplicaciones de visión por computador en tiempo real: estructuras de datos, procesamiento y análisis de imágenes, análisis estructural, etc. Este marco de

[12] Arduino IDE lo podemos encontrar en https://www.arduino.cc/en/softwarehttps://www.arduino.cc/en/software (último acceso: 22-06-2023)

trabajo facilita en gran manera el aprendizaje e implementación de distintas técnicas de visión por computador.

La librería OpenCV está dirigida fundamentalmente a la visión por computador en tiempo real. Entre sus muchas áreas de aplicación destacarían: interacción hombre-máquina, segmentación y reconocimiento de objetos, reconocimiento de gestos, seguimiento del movimiento, estructura del movimiento, y robots móviles [16].

OpenCV (Open Source Computer Vision Library) tiene una API[13] (Application Programming Interface) que proporciona una interfaz de programación para acceder a las funcionalidades de la biblioteca. La API de OpenCV está disponible en varios lenguajes de programación, incluyendo C++, Python, Java y MATLAB.

3.3 Puntos de referencia faciales con DLIB

Dlib contiene una amplia gama de algoritmos de aprendizaje automático. Todo diseñado para ser altamente modular, rápido de ejecutar y simple de usar a través de una API de C++ limpia y moderna. Se utiliza en una amplia gama de aplicaciones que incluyen robótica, dispositivos integrados, teléfonos móviles y grandes entornos informáticos de alto rendimiento [17].

Dlib es un detector facial de puntos de referencia con modelos pre-entrenados, Dlib se usa para estimar la ubicación de 68 coordenadas (x, y) que mapean los puntos faciales en la cara de una persona. Este usa un archivo con el modelo pre-entrenado «*shape_predictor_68_face_landmarks.dat*» [18].

[13] API OpenCV lo podemos encontrar en https://docs.opencv.org/4.x/d0/d1e/gapi.html (último acceso: 22-06-2023)

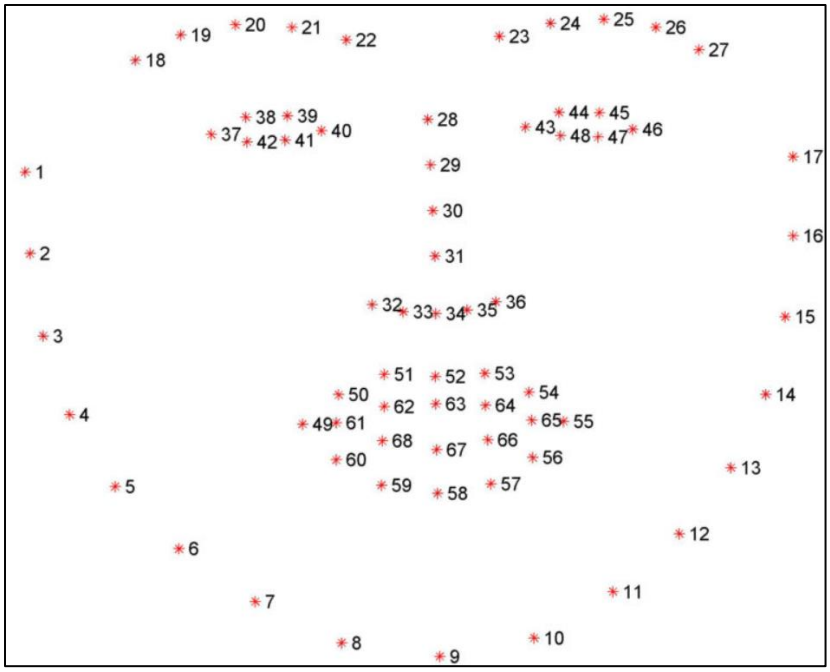

Figura 3-1 Predictor 68 face

3.3.1 Face-recognition

Sirve para reconocer rostros desde Python o desde la línea de comandos con la biblioteca de reconocimiento facial más simple. Creado con el reconocimiento facial de última generación de dlib y construido con aprendizaje profundo.

El modelo tiene una precisión del 99.38% en puntos de referencia de caras etiquetadas. También proporciona una sencilla herramienta de línea de comandos de reconocimiento de rostros que permite hacer reconocimiento facial desde una carpeta de imágenes [19].

3.3.2 Diferencia entre dlib y Face-recognition

Dlib es una biblioteca más amplia y completa que abarca una amplia gama de algoritmos y técnicas de visión por computadora, mientras que Face Recognition es una biblioteca especializada que se basa en dlib para ofrecer una interfaz fácil de usar y funciones específicas para el reconocimiento facial.

Face Recognition simplifica el proceso de reconocimiento facial al proporcionar funciones predefinidas y modelos entrenados, lo que facilita su implementación en proyectos de reconocimiento facial.

3.3.3 Conjunto de datos dlib

Utiliza modelos entrenados que se basan en conjuntos de datos populares, como el conjunto de datos "Labeled Faces in the Wild" (LFW) y el conjunto de datos "Helen".

Conjunto de datos Labeled Faces in the Wild (LFW): El conjunto de datos LFW contiene miles de imágenes de rostros de personas reales en diversas condiciones y se utiliza ampliamente en el campo del reconocimiento facial. Puedes encontrar más información y descargar el conjunto de datos desde el siguiente enlace: http://vis-www.cs.umass.edu/lfw/

Conjunto de datos Helen: El conjunto de datos Helen es un conjunto de imágenes faciales etiquetadas que se utiliza para entrenar modelos de puntos de referencia faciales. Contiene imágenes de rostros con anotaciones precisas de puntos de referencia faciales. Puedes obtener más información y descargar el conjunto de datos desde el siguiente enlace: http://www.ifp.illinois.edu/~vuongle2/helen/

3.4 Base de datos MongoDB

MongoDB es un sistema de gestión de base de datos no relacional. Lo que MongoDB almacena son objetos (a los que se les llama documentos) en lugar de filas. El formato de almacenamiento de objetos es del estilo de JSON [20].

JSON (JavaScript Object Notation) es un formato estandarizado para representación de objetos, muy utilizado en aplicaciones web [20]. Es un mecanismo de representación de objetos estandarizados por ECMA (European Computer Manufacturers Association) y ampliamente utilizado [20].

En conclusión, un objeto JSON es una colección no ordenada de pares (nombre, valor), un objeto empieza y termina respectivamente por {}; entre medias se colocan los pares separados por comas {“nombre”: “Andrés”}.

En la figura 3-2 se muestra una clase Cliente y la representación de JSON de un objeto de esta clase. El id es de tipo numérico (el valor no va entrecomillas), mientras que el resto son de tipo cadena [20].

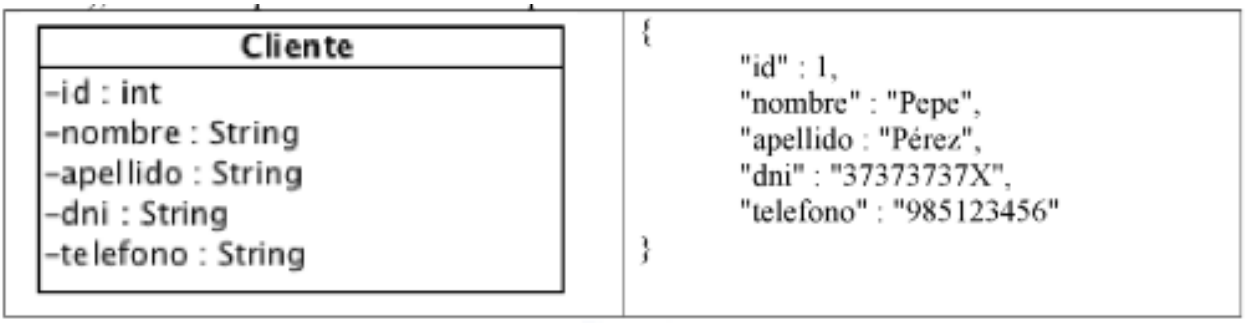

Figura 3-2 Ejemplo JSON

Los tipos de datos permitidos se resumen en la siguiente imagen.

Tipo de dato	Ejemplo
String(en unicode)	"nombre":"Andres"
int	"edad":30
real	"salario":1200,50
potencia de 10	"atomos":165e6
Array	"telefonos":["1234","6789"]

Objeto	"amigo": {
	"id": 1,
	"nombre": "Andres",
	"apellido":"Mendoza"
	"dni": "1234567S,
	"telefono":"698753321"
	}

Boolean	"permitido": true
null	"proyecto": null

Tabla 3-1. Tipos de datos

3.5 Protocolo WebSockets

A diferencia de HTTP, los WebSockets establecen una conexión bidireccional persistente entre el cliente y el servidor. A través de esta conexión, el cliente puede

enviar datos al servidor en cualquier momento, y el servidor puede enviar datos al cliente por iniciativa propia en cualquier momento. [21].

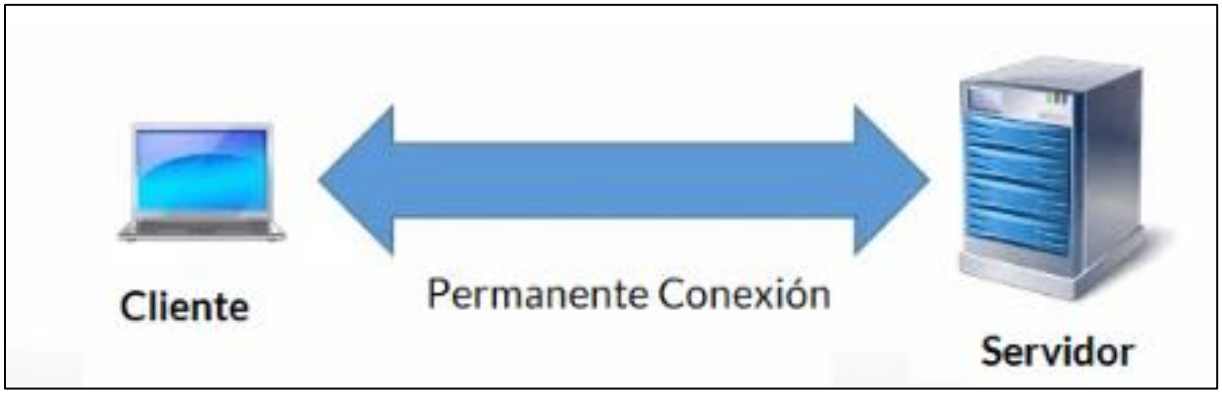

Figura 3-3 Conexión permanente entre cliente y servidor

3.6 Imágenes base64

Para que él envió seas más rápido las imágenes son convertidas en datos binarios; en la configuración que se realiza en el código cargado a la placa ESP32-CAM se realiza el siguiente proceso:

Las imágenes convertidas en binarios son troceadas en cinco partes antes de ser enviadas al servidor, una vez las imágenes llegan al servidor sockets son procesadas pares convertirlas en una sola imagen, por esta razón, primero se debe convertir la imagen a un formato binario (Base64)[14] para ser procesadas.

[14] Base64 lo podemos encontrar en https://www.base64-image.de/ (último acceso: 19-06-2023)

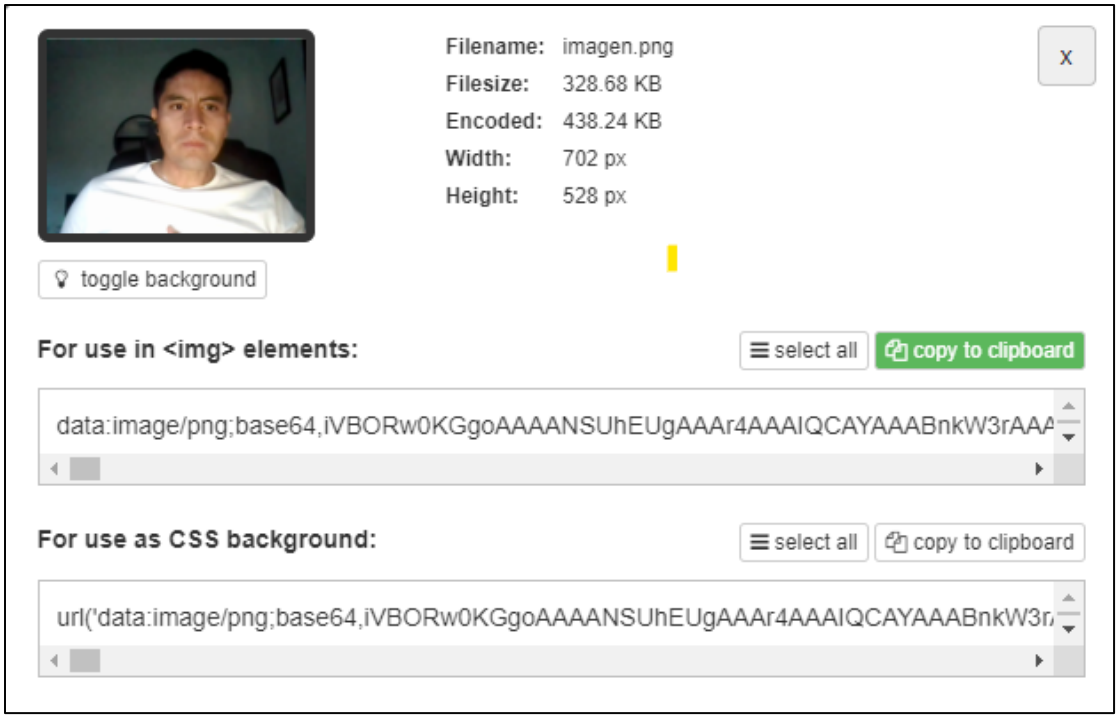

Figura 3-4 Imagen base64

3.7 Ultramsg

Ultramsg es una API multifuncional para WhatsApp y la mejor herramienta para empresas y programadores. Se puede integrar en cualquier sistema para contabilizar clientes, en CRM, ERP o en sitios web para enviar mensajes, notificar a los usuarios y muchos más [22].

Figura 3-5 Ultramsg

3.8 Aplicación Web

La aplicación Web se implementó en Php Codeigniter con el framewrok Bootstrap 5.

3.8.1 Php Codeigniter

CodeIgniter es un framework de desarrollo de aplicaciones web de código abierto y basado en PHP. CodeIgniter proporciona una estructura ligera y sencilla para desarrollar aplicaciones web rápidas y eficientes, siguiendo el patrón de diseño MVC (Modelo-Vista-Controlador). Su objetivo es permitir que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, brindando un conjunto de bibliotecas para tareas comunes, así como una interfaz simple y una estructura lógica para acceder esas bibliotecas. [23]

3.8.2 Framework Bootstrap 5

Bootstrap 5 es una de las versiones recientemente lanzadas del Framework Bootstrap para el desarrollo web. Esta es una de las librerías más conocidas, pues se pueden construir aplicaciones web adaptables para móvil con el CDN de *open source* jsDelivr y una página con una plantilla de inicio. [24]

Capítulo 4 - Implementación del sistema completo de IoT

En este capítulo se describe la implementación e integración de las diferentes tecnologías que se ha utilizado para el desarrollo del sistema y la arquitectura de la misma.

4.1 Arquitectura del sistema

La arquitectura del sistema propuesto incluye el dispositivo ESP32-CAM como el componente principal para la detección de somnolencia, la conexión WiFi para la transmisión de datos, un servidor para procesar los datos, una base de datos para almacenarlos y una interfaz de usuario para interactuar con el sistema.

Dispositivo ESP32-CAM actúa como el dispositivo de adquisición de imágenes y detección de somnolencia. Está equipado con una cámara para capturar imágenes en tiempo real y realizar el procesamiento necesario para detectar signos de cansancio en el conductor. Utiliza algoritmos de visión por computadora y técnicas de procesamiento de imágenes para analizar las imágenes capturadas y determinar el nivel se somnolencia.

Servidor actúa como punto central para recibir los datos del ESP32-CAM. Recibe las imágenes capturadas y los resultados de la detección de somnolencia a través de una conexión WiFi y almacena en la base de datos para su posterior análisis.

La base de datos almacena los datos recopilados, incluyendo las imágenes capturadas, los resultados de la detección de somnolencia y cualquier otro dato relevante. Estos datos serán utilizados para el análisis posterior, generación de informes, comparación de datos a lo largo del tiempo y otras tareas relacionadas.

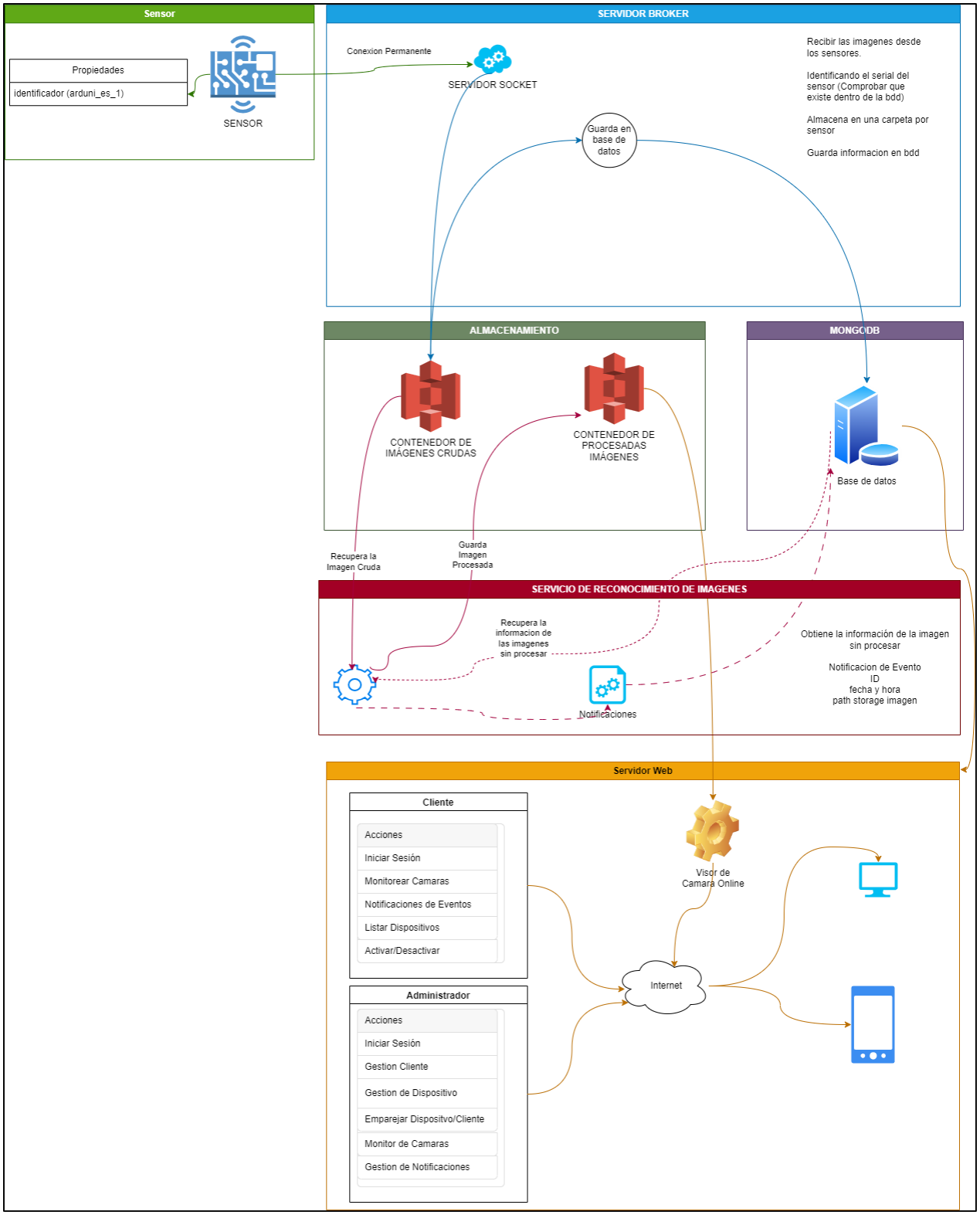

Figura 4-1 Arquitectura DDD

4.2 Setup para la implementación del algoritmo DDD

Para la implementación del desarrollo del algoritmo del DDD (Driver Drowsiness Detection), se ha implementado diferentes librerías.

4.2.1 Librerías utilizadas

En la raíz del proyecto tenemos un archivo **"requirements.txt"** con todas las librerías que requiere el proyecto. Luego abrir la ruta del proyecto en un terminal y digitar el siguiente comando para instalar las librerías con el siguiente comando:

python -m pip install -r requirements.txt

4.2.2 Instalación DLIB

En el siguiente apartada se explica la correcta instalación de la biblioteca dlib para un Sistema Operativo Windows.

4.2.2.1 Instalación dependencias C++

Empezamos instalando las dependencias C++, para ello tendremos que descargar el instalador de Visual Studio 2022[15], que es la última versión por ahora. Nos dirigimos a "Community" y lo descargamos.

[15] Visual Studio 2022 lo podemos encontrar https://visualstudio.microsoft.com/es/downloads/ (último acceso: 10-06-2023)

Figura 4-2 Visual Studio 2022

En la siguiente ventana elegimos "Desarrollo para escritorio con C++" y luego seleccionamos instalar.

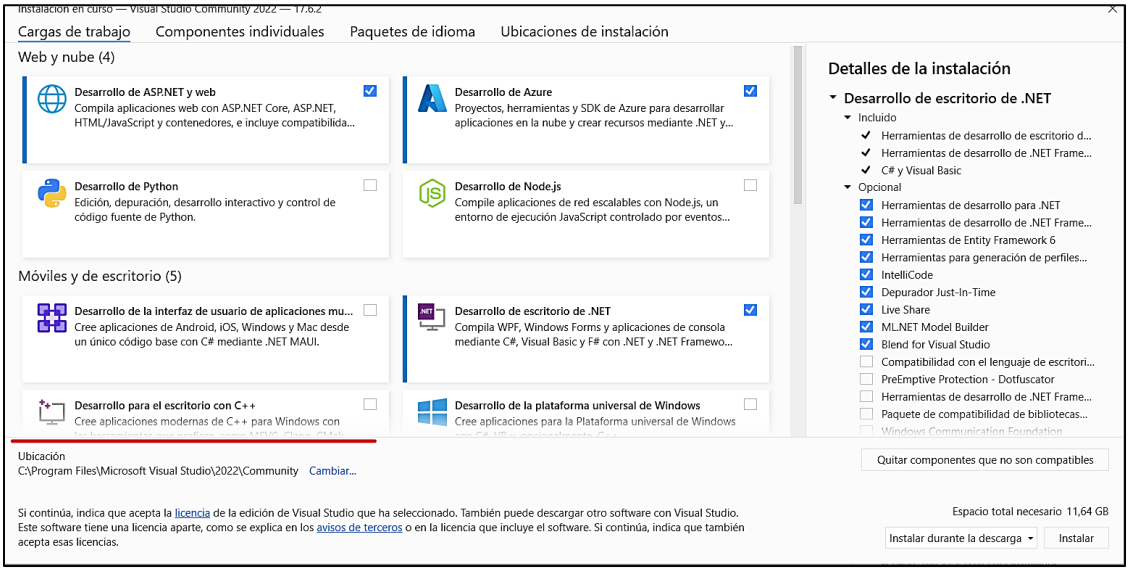

Figura 4-3Dependencia C++

4.2.2.2 Instalación Cmake

Abrir la ruta de la carpeta del proyecto y en el terminal digitar: pip install cmake

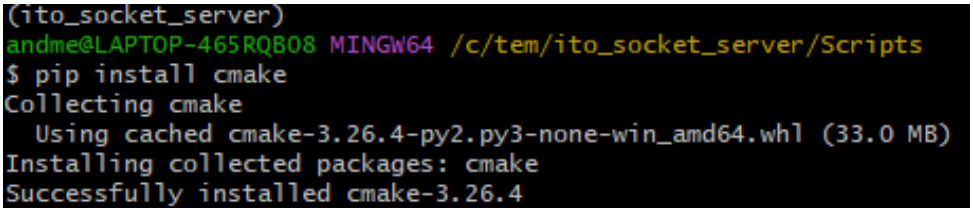

Figura 4-4 Libreria cMake

4.2.2.3 Instalación librería dlib

Abrir la ruta de la carpeta del proyecto y en el terminal digitar: pip install dlib

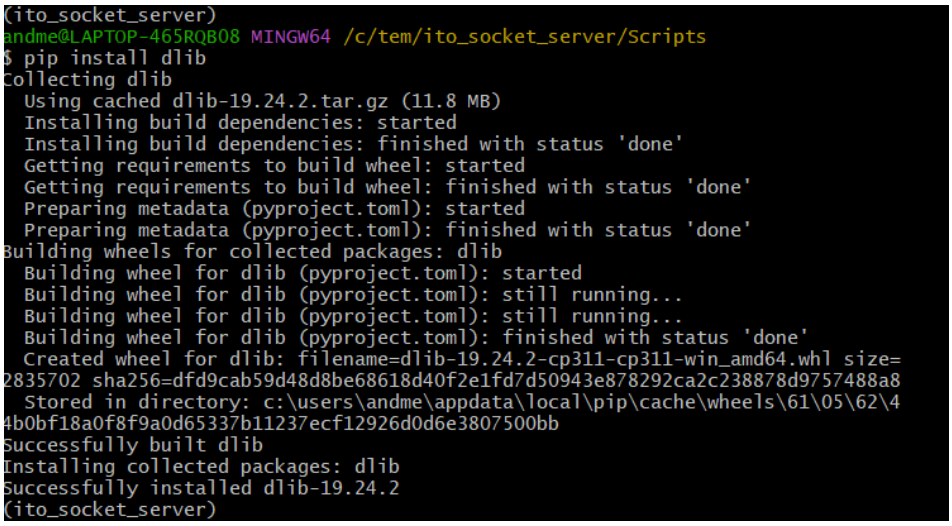

Figura 4-5 Librería dlib

4.2.2.4 Testeo de correcta instalación Sistema Operativo Windows

Para que la instalación dlib sea exitosa primero debemos verificar que versión de Python que tenemos. La recomendada es la 3.10.6.

Una vez verificada la versión debemos digitar el siguiente comando para mirar las versiones coincidentes con tu versión de Python: **pip debug –verbose.**

A continuación, descargar el paquete que corresponda a tu versión desde la siguiente ruta dlib-cp310[16], añadir en la raíz del proyecto el paquete descargado y por último abrir un terminar en la ruta del proyecto y digitar el siguiente comando: **pip install dlib-19.22.99-cp310-cp310-win_amd64.whl.**

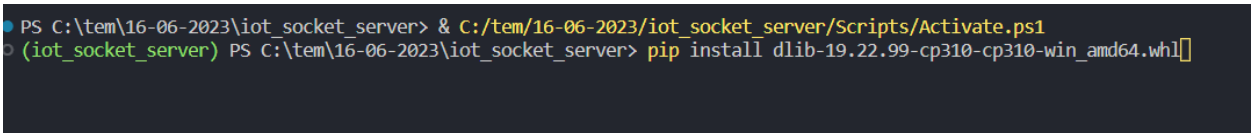

Figura 4-6 dlib para SO Windows

4.3 Descripción del sistema desarrollado

En este apartado se describe la implementación de toda la arquitectura del sistema propuesto.

4.3.1 Algoritmo cargado en la placa ESP32-CAM

La conexión WiFi en un ESP32-CAM es un aspecto fundamental y esencial para el correcto funcionamiento de la cámara y su capacidad de comunicación inalámbrica, lo que brinda acceso a Internet y la posibilidad de comunicarse con otros dispositivos en la red.

La resolución de una imagen se refiere a la cantidad de pixeles que componen la imagen. En general, una imagen de alta resolución contiene más detalles y ofrece una mayor precisión en la representación visual de objetos y características. Para nuestro proyecto hemos utilizado una resolución alta ya que se requiere un análisis

[16] EL paquete dlib-cp310 lo podemos encontrar en https://github.com/datamagic2020/Install-dlib/blob/main/dlib-19.22.99-cp310-cp310-win_amd64.whl (último acceso: 20-05-2023)

detallado se la imagen, como la identificación de rasgos faciales para la detección de la persona y somnolencia.

El ESP32-CAM realiza la captura de la imagen, convierte en formato Base64, corta la imagen en segmentos y solicita la conexión al servidor sockets para luego enviar las imágenes segmentadas por los cinco puertos configurados que constantemente están enviando las imágenes al servidor.

Las imágenes son datos binarios, es decir, una secuencia de bytes. Los sockets generalmente trabajan con datos en forma de cadenas de caracteres. Para poder transmitir una imagen a través de un socket, es necesario convertir los datos binarios de la imagen en una cadena de caracteres que pueda ser enviada y recibida correctamente.

Al transmitir datos binarios a través de redes, es posible que algunos caracteres o secuencias de bytes se interpreten incorrectamente o se modifiquen debido a las conversiones y protocolos utilizados en la transmisión. La conversión a Base64 garantiza que los datos binarios se representen de manera segura y compatible con los protocolos de comunicación, ya que utiliza un conjunto limitado de caracteres seguros y ampliamente reconocidos.

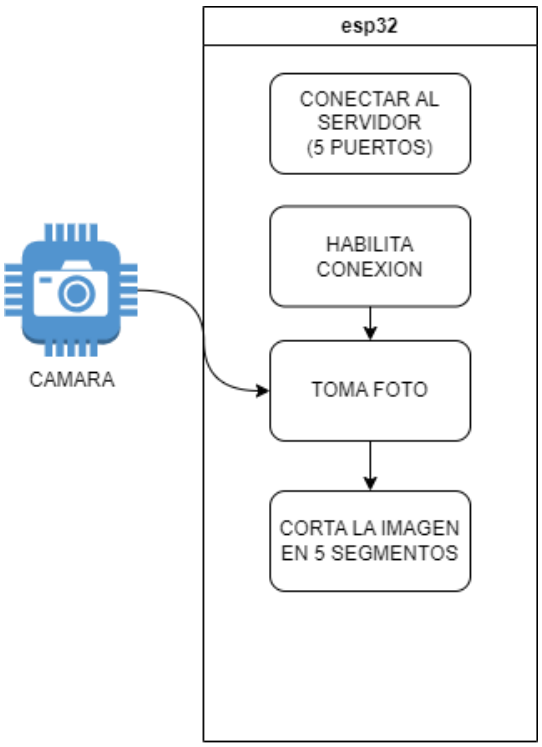

Figura 4-7 Estructura del Algoritmo para la placa ESP32-CAM

4.3.2 Algoritmo de somnolencia

A continuación, detallamos los puntos de referencia del detector ***«shape_predictor_68_face_landmarks.dat»*** utilizado para la detección de rasgos faciales en imágenes.

```
shape[34],     # Nose tip
shape[9],      # Chin
shape[46],     # Left eye left corner
shape[37],     # Right eye right corne
shape[55],     # Left Mouth corner
shape[49]      # Right mouth corner
```

Figura 4-8 Puntos de Referencia

La biblioteca **face-recognition** [25] permite detectar y localizar rostros faciales en imágenes cargadas en un directorio local o servidor. Además, compara las

características de los rostros y determina la similitud entre ellas para dar un valor de verdadero o falso.

El algoritmo de somnolencia obtiene el ultimo registro de la base de datos (imagen cruda), procesa la imagen, guarda la imagen y actualiza el registro enviando la imagen procesada a la base de datos.

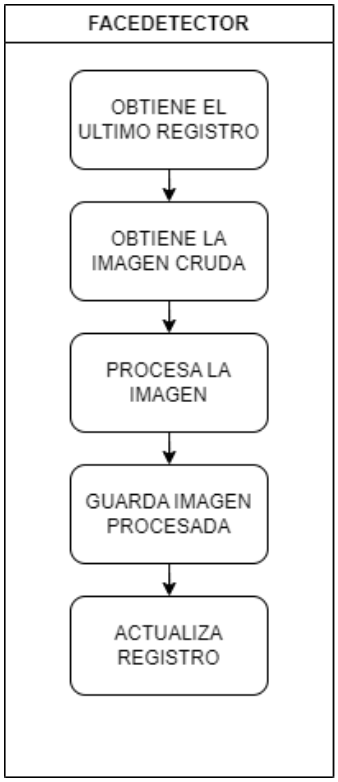

Figura 4-9 Estructura del algoritmo de somnolencia

4.3.3 Servidor sockets

Recibe la solicitud de conexión por parte del ESP32-CAM para habilitar y conceder la conexión de los cinco puertos configurados.

Una vez concedida la conexión recibe los segmentos de las imágenes capturadas por la cámara del ESP32-Cam.

Estas imágenes son procesadas para crear una sola imagen y guardarla en el disco, para finalmente agregar el registro en el disco de cada imagen capturada.

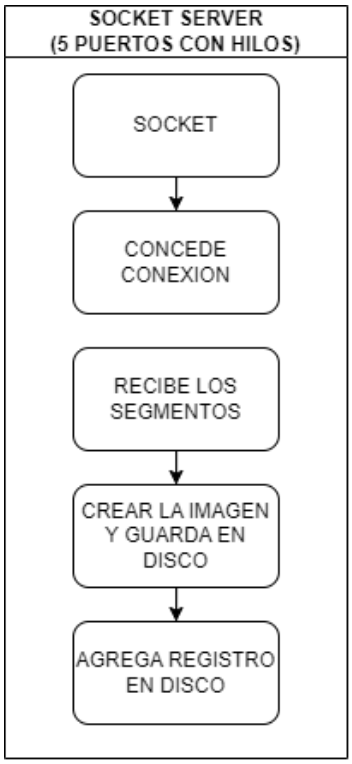

Figura 4-10 Estructura del servidor sockets

4.3.4 Base de datos MongoDB

La base de datos para el proyecto se compone de seis colecciones(tablas). En la Figura 4-11 se presenta la estructura general de la base de datos del sistema que fue implementada en MongoDB.

La estructura de la base de datos MongoDB está orientada a documentos, lo que significa que los datos se almacenan en documentos flexibles y sin esquema fijo.

La colección "Asignación" se encuentra relacionada con las colecciones device y cliente la cual se encarga de traer la información de las colecciones relacionadas. Además de verificar la asignación de un dispositivo al usuario.

La colección "Cliente" se encuentra relacionada con la colección empleado y se encarga de almacenar la información de los datos del cliente y a su vez obtener la información de la colección empleado.

La colección "Device" se encarga de almacenar la información de cada dispositivo y enviar la información a la colección asignación.

La colección "tracking" registra y almacena información relacionada con el seguimiento de los eventos. Su principal función es mantener un historial de las transiciones o actualizaciones de estado de elementos específicos para enviar la información a la colección notificaciones.

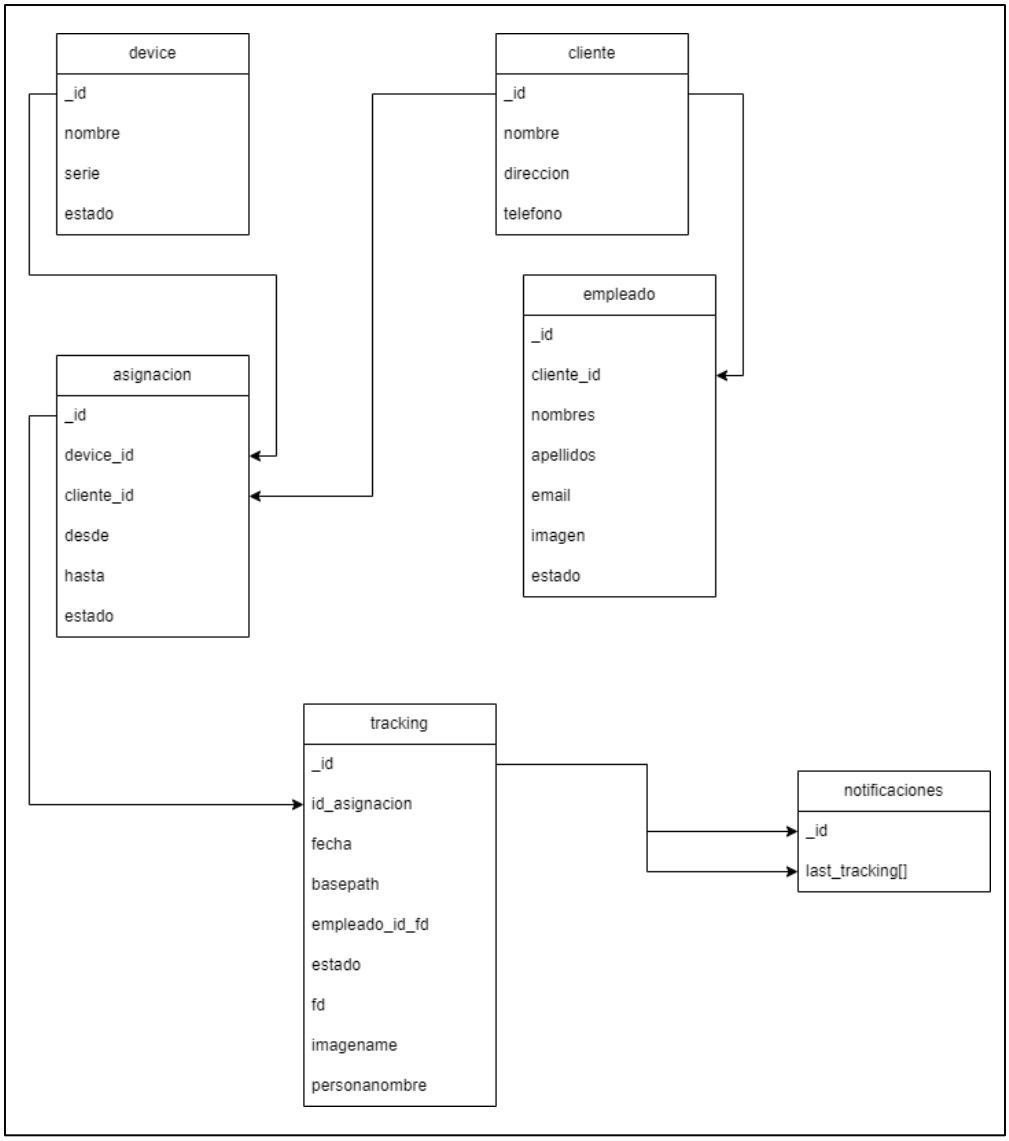

Figura 4-11 Estructura general de la base de datos MongoDB

4.3.5 Notificaciones mediante Ultramsg

Las notificaciones de los estados de somnolencia se envían de forma remota al dispositivo móvil. Se configura el número de móvil en ultramsg y desde ese número llega la notificación a los usuarios.

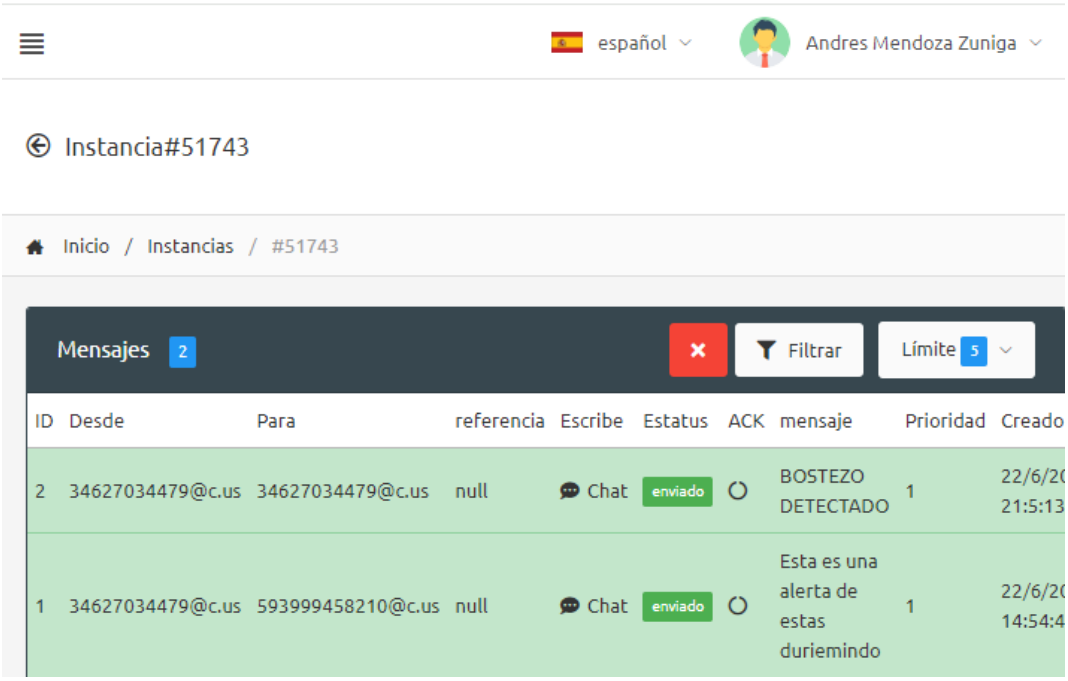

Figura 4-12 Notificación de somnolencia

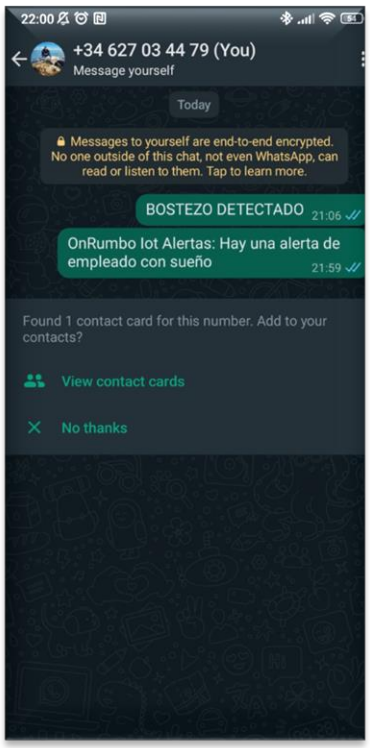

Figura 4-13 Notificación de Somnolencia móvil

4.3.6 Desarrollo Aplicación WEB

La aplicación web para el sistema de detección de síntomas de cansancio en conductores está compuesta por varios componentes para brindar una experiencia completa y funcional a los usuarios. A continuación, presento la estructura y los componentes clave de la aplicación web.

4.3.6.1 Página de inicio

Es la primera visita de los usuarios al acceder a la aplicación web. Donde incluirá la descripción del sistema y sus beneficios, así como la opción para iniciar sesión.

4.3.6.2 Registro y autenticación de usuarios

En esta sección de la aplicación permitirá autenticarse para acceder a sus perfiles y utilizar las funcionalidades de la aplicación.

4.3.6.3 Perfil de usuario

En esta sección cada usuario va a tener su perfil personalizado donde podrían ver y editar su información personal, como nombre, dirección de correo, etc. Donde podrán acceder a su historial de eventos de somnolencia y resultados.

4.3.6.4 Dispositivos

En esta sección la aplicación permitirá a los usuarios verificar los dispositivos ESP32-CAM que tienen asignados a su perfil.

4.3.6.5 Historial y estadísticas

En esta sección permitirá a los usuarios acceder a su historial de eventos de somnolencia, resultados de pruebas anteriores y estadísticas relacionadas. Los usuarios podrán ver gráficos que resumen el rendimiento a lo largo del tiempo, lo que les permitirá hacer un seguimiento de su fatiga y tomar medidas para mejorar su seguridad en la conducción.

4.4 Tecnologías para la gestión del proyecto

Las tecnologías que se ha implementado para el proyecto son:

- **GitHub[17]:** es un sistema de control de versiones que es utilizado para el control de versiones que se va obteniendo en el desarrollo del aplicativo.

[17] El código implementado lo podemos encontrar en https://github.com/andmendoza/TFM-IoT (último acceso: 22-05-2023)

Capítulo 5 - Experimentación y a ajuste del sistema

5.1 Configuración ESP32-CAM en Arduino IDE

La primera fase de evaluación consistió en comprobar la conexión exitosa al servidor y el envío correcto de las imágenes capturadas por el ESP32-CAM al servidor.

```
CAMARA OK
http://192.168.1.176/cam-lo.jpg
http://192.168.1.176/cam-hi.jpg
-- Obtiene Imagen
-- Imagen Segmentada
Inciciado socket ->
conectado al servidor
conectado al servidor
conectado al servidor
conectado al servidor
conectado al servidor
.....
<- Finalizado envio socket
```

Figura 5-1 Conexión al servidor

El envió de imágenes mediante sockets permite la conexión en tiempo real, al usar un solo puerto la imagen tiene un tiempo de envío largo o incluso no se llega a enviar la imagen; se validó el envío de la imagen hasta con 10 puertos y se comprobó que el ESP32-CAM no soportaba y se reiniciaba el dispositivo. Por esta razón se pudo validar que el número ideal del envío de la imagen es por cinco puertos que mejoro el tiempo de envió con hasta 2 imágenes por segundo.

	Envío de imágenes por puertos									
	Puertos									
	Uno	Dos	Tres	Cuatro	Cinco	Seis	Siete	Ocho	Nueve	Diez
frame/segundos	No envía imagen	frame/ 16s	frame/ 8s	frame/ 3s	frame/ 1,33s	Error	Error	ESP32-CAM no soporta	ESP32-CAM no soporta	ESP32-CAM no soporta

Tabla 5-1. Envió de imagen por puertos

El tiempo de procesamiento de las imágenes es de milésimas de segundo dependiendo del procesador del ordenador.

Para la parte del reconocimiento de rostros el tiempo de procesamiento es alrededor de cuatro milésimas de segundo.

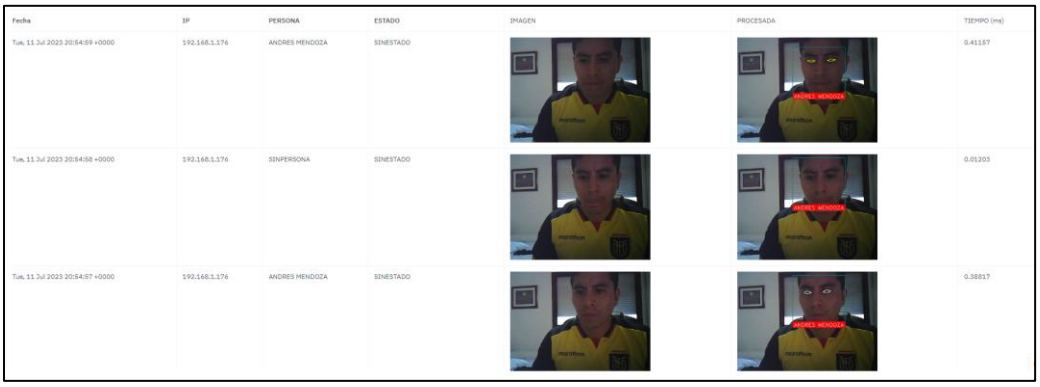

Figura 5-2 Tiempo de procesamiento

5.2 Servidor Sockets

La segunda fase de evaluación consiste en verificar que el servidor se encuentra constantemente recibiendo los datos de las imágenes enviadas por los sensores ESP32-CAM, el servidor identifica el serial de cada sensor comprobando si existe dentro de la base de datos para almacenar cada imagen en una carpeta por sensor.

En la Figura 5-3 podemos verificar que las imágenes recibidas se almacenan en una carpeta del servidor por cada sensor ESP32-CAM.

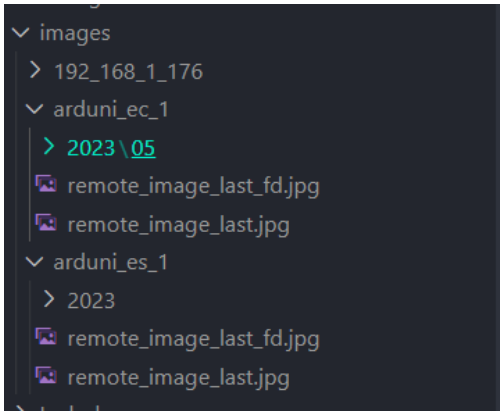

Figura 5-3 Carpeta de almacenamiento de imágenes

En la Figura 5-4 podemos verificar que la información de las imágenes recibidas se guarda correctamente en la base de datos.

arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:43.681	20230523150043681991.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:43.817	20230523150043817904.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:44.029	20230523150044029506.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:44.235	20230523150044235320.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:44.392	20230523150044392347.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:44.571	20230523150044571832.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:44.749	20230523150044749633.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:45.017	20230523150045017602.jpg	192.168.200.15		(Document) 8 Fields
arduni_ec_1/2023/05/23/15/00/			0	2023-05-23 15:00:45.185	20230523150045185419.jpg	192.168.200.15		(Document) 8 Fields
arduni_es_1/2023/06/11/18/25/	(N/A)		0	2023-06-11 18:25:01.666	20230611182501666314.jpg	192.168.1.176	(N/A)	(Document) 8 Fields
arduni_es_1/2023/06/11/18/25/	(N/A)		0	2023-06-11 18:25:02.642	20230611182502642998.jpg	192.168.1.176	(N/A)	(Document) 8 Fields
arduni_es_1/2023/06/11/18/25/	(N/A)		0	2023-06-11 18:25:04.105	20230611182504105257.jpg	192.168.1.176	(N/A)	(Document) 8 Fields
arduni_es_1/2023/06/11/18/25/	(N/A)		0	2023-06-11 18:25:04.885	20230611182504885412.jpg	192.168.1.176	(N/A)	(Document) 8 Fields
arduni_es_1/2023/06/11/18/25/	(N/A)		0	2023-06-11 18:25:05.903	20230611182505903254.jpg	192.168.1.176	(N/A)	(Document) 8 Fields

Figura 5-4 Información enviada por el servidor sockets

Se realiza pruebas con Api RESTFUL y Sockets, se verifica que lo apropiado es utilizar sockets en lugar de una API RESTFUL por las siguientes razones:

- El ESP32-CAM como WebServer es inaccesible desde redes externas.

- El uso de REST API para el envío de imágenes tuvo un tiempo de envío aproximado de 5 o más segundos entre conectar, enviar y finalizar el proceso de envío de cada imagen.

5.3 Servidor de Inferencia dlib

La tercera fase de evaluación consiste en verificar la correcta implementación del algoritmo DDD.

En la Figura 5-5 se puede observar la imagen real y la procesada en la cual se puede verificar el éxito de la implementación del algoritmo mediante la librería face-recognition.

Fecha	IP	PERSONA	ESTADO	IMAGEN	PROCESADA
Fri, 16 Jun 2023 16:05:04 +0000	192.168.1.176	SINPERSONA	SINESTADO		

Figura 5-5 Reconocimiento de la persona

En la Figura 5-6 y Figura 5-7 se puede identificar que realiza el proceso correcto de los patrones asociados con la somnolencia, como el parpadeo de los ojos y el bostezo.

Fecha	IP	PERSONA	ESTADO	IMAGEN	PROCESADA
Sun, 11 Jun 2023 19:59:04 +0000	192.168.1.176	ANDRES MENDOZA	SINESTADO		

Figura 5-6 Reconocimiento de parpadeo

Fecha	IP	PERSONA	ESTADO	IMAGEN	PROCESADA
Sun, 11 Jun 2023 19:56:29 +0000	192.168.1.176	ANDRES MENDOZA	BOSTEZO DETECTADO		

Figura 5-7 Reconocimiento de bostezo

En la Figura 5-8 se observa el procesamiento de cada imagen por el servidor de inferencia.

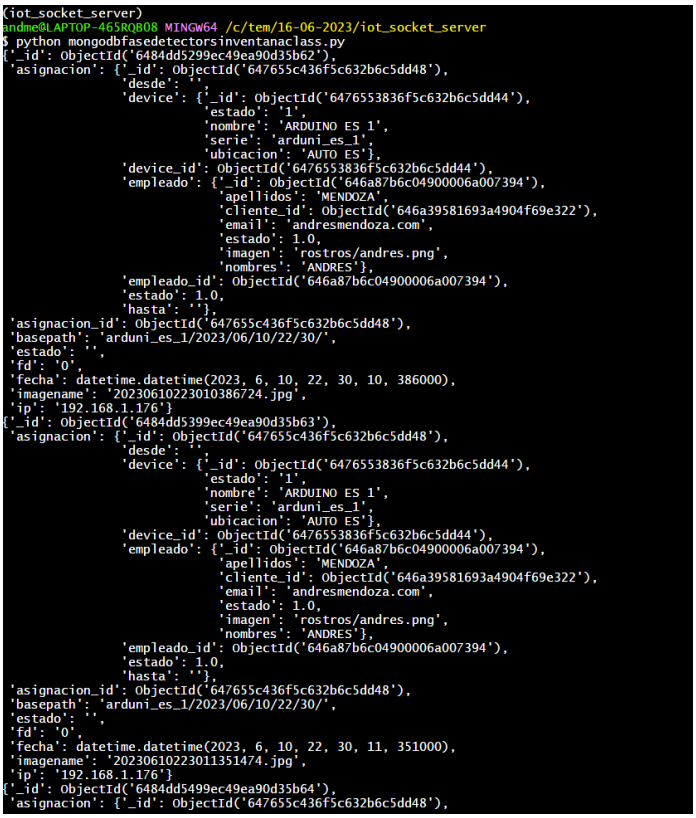

Figura 5-8 Información del proceso en formato JSON

5.4 Base de datos MongoDB

MongoDB ofrece una variedad de opciones de configuración que permite ajustar el comportamiento del sistema, como el tamaño del cache, tiempo de espera de las conexiones y la cantidad máxima de conexiones simultaneas. Según estos factores se realiza una configuración acorde a las necesidades del sistema.

El servidor almacena físicamente como imágenes las capturas realizadas por el ESP32-CAM y en la Figura se puede verificar que almacena correctamente la ruta donde se encuentra las imágenes capturadas.

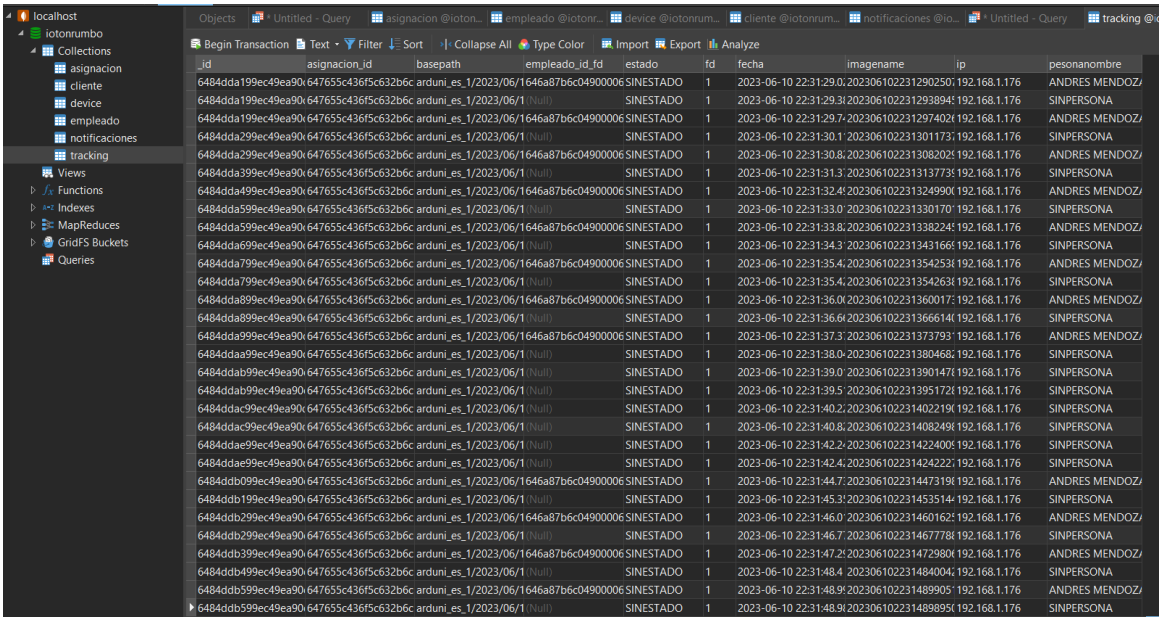

Figura 5-9 Ruta cargados en la BBDD

5.5 Aplicación WEB

La quinta etapa verifica que los datos se extraen correctamente de la base de datos y que todos sus componentes brindar una experiencia completa y funcional a los usuarios.

5.5.1 Login

Proceso de autenticación que permite al usuario acceder al sistema de detección de somnolencia.

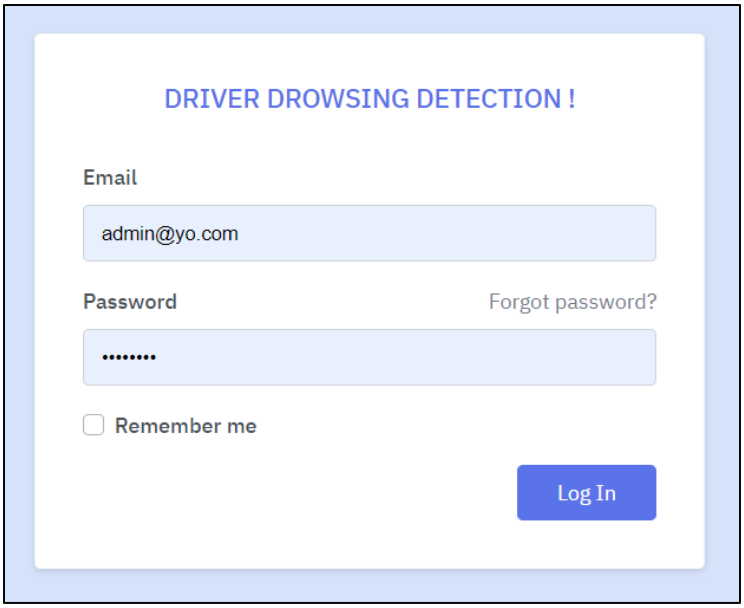

Figura 5-10 Login

5.5.2 Dashboard

Parte visual de los datos obtenidos de cada dispositivo ESP32-CAM

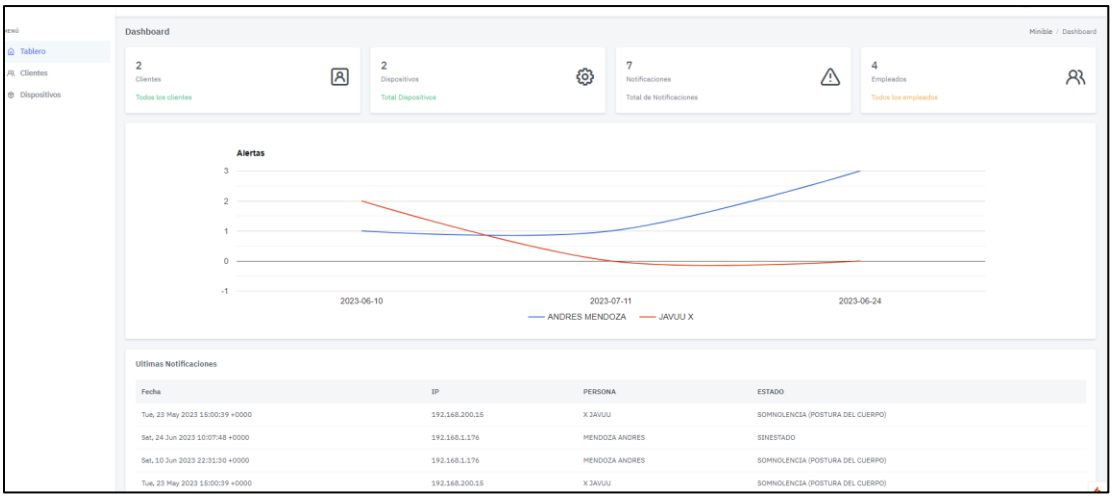

Figura 5-11 Dashboard

5.5.3 Clientes

En la Figura 5-12 se puede observar los clientes registrados en la aplicación.

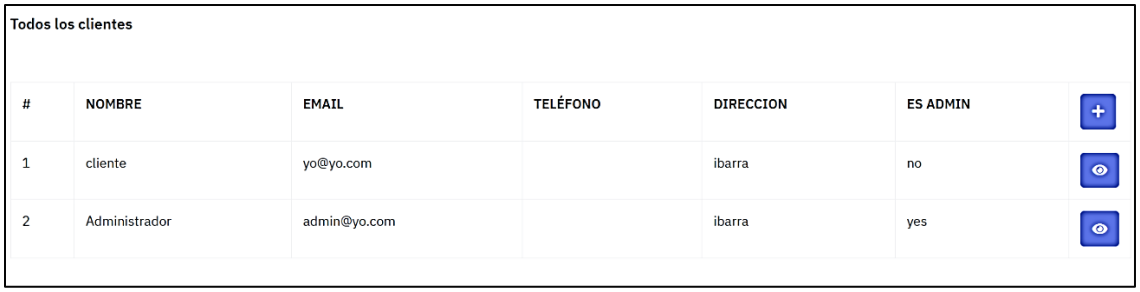

Figura 5-12 Clientes

Pulsando en el botón Añadir "icono cruz" podemos añadir usuarios/clientes.

Figura 5-13 Añadir usuarios/clientes

Pulsando en el botón editar "icono ojo" podemos editar los campos obligatorios.

Figura 5-14 Editar usuario/cliente

Pulsando en el botón Empleados podemos verificar las imágenes que se encuentran cargadas en el servidor y que serán identificadas por el algoritmo mediante el face-recognition

Figura 5-15 Empleados

Pulsando en el botón Asignaciones se verifica el dispositivo que se encuentra asignado a cada conductor.

Figura 5-16 Asignaciones

- En el botón LOG TRACKING podemos verificar el proceso de cada imagen procesado por fecha, hora y la IP asignada a cada dispositivo.

Figura 5-17 Tracking

- En el botón REMOTO podemos verificar el proceso de captura de cada imagen efectuada por el ESP32-CAM.

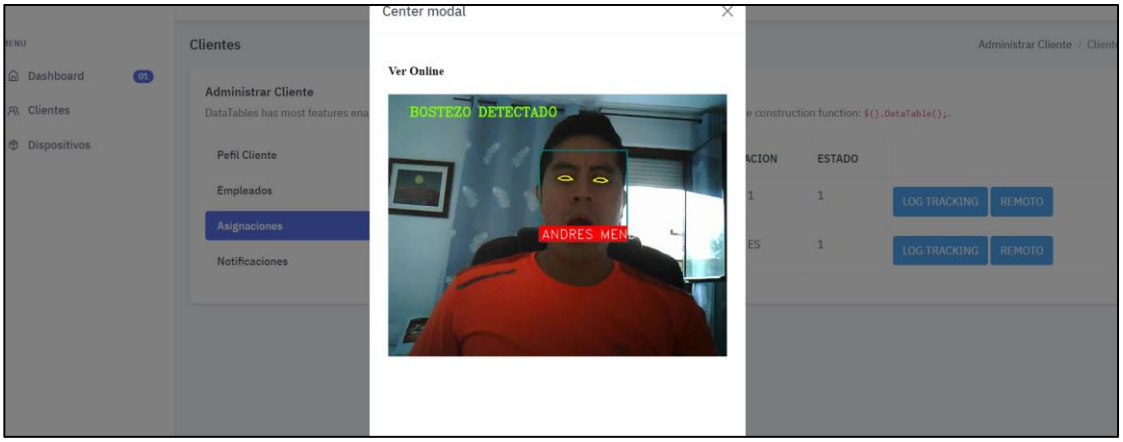

Figura 5-18 Visor Online

Pulsando en el botón notificaciones se verifica el dispositivo las notificaciones enviadas que fueron detectadas por el algoritmo de somnolencia.

Figura 5-19 Notificaciones

5.5.4 Dispositivos

En la Figura 5-20 se puede observar los dispositivos registrados que se encuentran añadidos a cada conductor.

Figura 5-20 Dispositivos ESP32-CAM

Pulsando en el botón Añadir “icono cruz” podemos añadir un nuevo dispositivo.

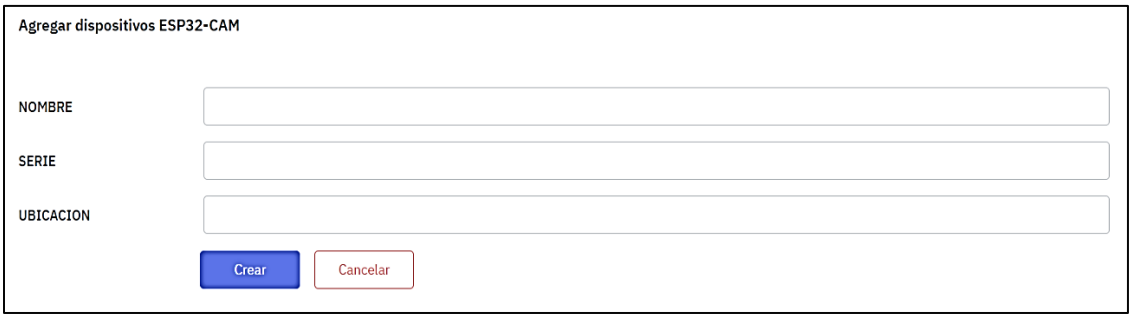

Figura 5-21 Añadir ESP32-CAM.

Capítulo 6 - Conclusiones y trabajo futuro

En conclusión, el desarrollo de una aplicación web relacionada con la somnolencia y el detector del cansancio utilizando el ESP32-CAM presenta varias consideraciones importantes:

- El estudio y familiarización de las librerías OpenCv y dlib proporcionaron las bases necesarias para desarrollar la aplicación de detección de somnolencia. Estas bibliotecas ofrecieron una amplia gama de funciones y algoritmos que permiten el procesamiento de imágenes y videos, así como la detección y seguimiento de rostros.

- Las notificaciones de somnolencia pueden interactuar de diferentes maneras dependiendo de las necesidades. Para el sistema de somnolencia se utilizó las notificaciones remotas configuradas en ultramsg que mediante una notificación al móvil informa sobre la detección de somnolencia en tiempo real.

- En consecuencia, a causa de recursos económicos no fue efectivo lograr la segmentación de usuarios para implementar el ESP32-CAM en flotas y esto representa una limitación significativa en la viabilidad y efectividad del sistema. No obstante, la implementación se puede adaptar adecuadamente a las necesidades y requisitos de cada tipo de conductor y flota.

- Con esta aplicación web se facilita a los usuarios que puedan visualizar y evidenciar todos los datos necesarios de somnolencia que han presentado los conductores durante el inicio de captura de imágenes obtenidas por el ESP32-CAM.

Para una siguiente versión del sistema configurado en el dispositivo ESP32-CAM sería conveniente Integrar un servicio MQTT en el servidor de reconocimiento de imágenes

que se conecte a un tópico de alerta y el ESP32-CAM se suscriba al tópico para emitir la alerta al conductor mediante una alarma sonora.

Además, la implementación del sistema de detección de somnolencia en la nube es completamente viable ya que al existir diversas empresas que ofrecen el servicio en la nube se podrá elegir la mejor alternativa y esto dependerá de varios factores, como la configuración exacta del sistema, la cantidad y el tiempo de recursos utilizados, la ubicación geográfica y el volumen de tráfico.

Chapter - Introduction

We are currently living in a constantly evolving technological age. Technological advances have had a significant impact on nearly every aspect of our lives, from the way we communicate and work to how we access information and entertain ourselves.

Technology has driven the development of areas such as artificial intelligence, the internet of things, robotics and augmented reality, among others. These emerging technologies are changing the way we interact with the world and opening up new possibilities in fields such as medicine, energy, transportation, and the environment.

However, along with the benefits, there are also challenges and responsibilities. It is important to address issues such as data security and privacy, the digital divide, and the environmental impact of technology. Ethics and regulation play a fundamental role in ensuring that technology is used responsibly and for the benefit of society as a whole.

In short, we are immersed in a technological age in which innovation and change are constant. Technology is transforming the way we live, work and interact, and it will continue to be a key factor in the future development of society.

For this reason, the objective of this work is to design a drowsiness detection system using the ESP32-CAM that is capable of identifying signs of drowsiness in a person, such as closing the eyes or yawning, through an intuitive and easy interface. to use that allows users feedback on their driving behavior when they experience drowsiness or fatigue.

Chapter - Conclusions and future work

In conclusion, the development of a web application related to sleepiness and fatigue detector using the ESP32-CAM presents several important considerations:

- The study and familiarization of the OpenCv and dlib libraries provided the necessary bases to develop the drowsiness detection application. These libraries offered a wide range of features and algorithms that enable image and video processing, as well as face detection and tracking.

- Sleepiness notifications can interact in different ways depending on the needs. For the drowsiness system, the remote notifications configured in ultramsg were used, which, by means of a notification to the mobile, informs about the detection of drowsiness in real time.

- Consequently, due to economic resources, it was not effective to achieve the segmentation of users to implement the ESP32-CAM in fleets and this represents a significant limitation in the feasibility and effectiveness of the system. However, the implementation can be adequately adapted to the needs and requirements of each type of driver and fleet.

- With this web application, users can view and demonstrate all the necessary sleepiness data that drivers have presented during the start of capturing the images obtained by the ESP32-CAM.

For a next version of the system configured in the ESP32-CAM device, it would be convenient to integrate an MQTT service in the image recognition server that connects to an alert topic and the ESP32-CAM subscribes to the topic to issue the alert to the driver through an audible alarm.

In addition, the implementation of the drowsiness detection system in the cloud is completely feasible since there are several companies that offer the service in the cloud, the best alternative can be chosen and this will depend on several factors, such as the exact configuration of the system, the amount and time of resources used, geographic location and volume of traffic.

Capítulo 7 - Bibliografía

[1] «ONDAVASCA,» 09 05 2022. [En línea]. Available: https://www.ondavasca.com/doctor-carlos-egea-presidente-federacion-de-sociedades-medicas-del-sueno-tendemos-a-pensar-que-el-sueno-es-una-perdida-de-tiempo-y-que-ese-tiempo-lo-tenemos-que-utilizar-para-otras-cosas/. [Último acceso: 17 06 2023].

[2] EuropaPress, «NIUS,» 2022. [En línea]. Available: https://www.niusdiario.es/sociedad/trafico/20220714/falta-sueno-comparable-conducir-tasa-alcoholemia_18_07014686.html. [Último acceso: 24 01 2023].

[3] F. MAPFRE, «Fundación MAPFRE,» 2021. [En línea]. Available: https://www.fundacionmapfre.org/en/education-outreach/road-safety/adas-systems/types/fatigue-detection-system/. [Último acceso: 24 01 2023].

[4] «SITRACK,» 16 08 2022. [En línea]. Available: https://blog.sitrack.com/que-es-el-sistema-adas-y-como-beneficia-a-los-vehiculos-de-tu-flota#:~:text=Beneficios%20de%20los%20sistemas%20ADAS&text=Adem%C3%A1s%20de%20la%20prevenci%C3%B3n%20de,vida%20%C3%BAtil%20de%20los%20veh%C3%ADculos.. [Último acceso: 14 07 2023].

[5] «Fundacion MAPFRE,» 2023. [En línea]. Available: https://www.seguridadvialenlaempresa.com/blog/como-funciona-el-sistema-de-deteccion-de-fatiga/. [Último acceso: 17 06 2023].

[6] BOSH, «Soluciones de movilidad de Bosch,» 2022. [En línea]. Available: https://www.bosch-mobility-solutions.com/en/solutions/assistance-systems/driver-drowsiness-detection/. [Último acceso: 24 01 2023].

[7] «Driver Drowsiness Detection,» 2015. [En línea]. Available: http://auto2015.bosch.com.cn/ebrochures2015/automated/cc/da/ddd/driver_drowsiness_detection_en.pdf. [Último acceso: 08 06 2023].

[8] «AWS,» 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/iot/#:~:text=El%20t%C3%A9rmino%20IoT%2C%20o%20Internet,como%20entre%20los%20propios%20dispositivos.. [Último acceso: 17 06 2023].

[9] «programarfacil,» 2023. [En línea]. Available: https://programarfacil.com/esp32/esp32-cam/. [Último acceso: 10 06 2023].

[10] «DescubreArduino,» 2023. [En línea]. Available: https://descubrearduino.com/sim800l-gsm/. [Último acceso: 18 06 2023].

[11] Lupita, «Programa en Línea,» 25 05 2022. [En línea]. Available: https://www.programaenlinea.net/razones-para-usar-mongodb-en-tus-proyectos/. [Último acceso: 2023 06 04].

[12] «wikipedia.org,» 17 12 2019. [En línea]. Available: https://es.wikipedia.org/wiki/Servidor. [Último acceso: 17 06 2023].

[13] «AWS,» 2023. [En línea]. Available: https://aws.amazon.com/es/what-is/web-application/#:~:text=Una%20aplicaci%C3%B3n%20web%20es%20un,y%20de%20una%20forma%20segura.. [Último acceso: 17 06 2023].

[14] «ARDUINO,» 2023. [En línea]. Available: https://support.arduino.cc/hc/en-us. [Último acceso: 04 06 2023].

[15] F. Zyprian, «Konfuzio,» 09 03 2023. [En línea]. Available: https://konfuzio.com/es/cv2/#:~:text=Python%20es%20un%20popular%20lenguaje,im%C3%A1genes%20y%20v%C3%ADdeo%2C%20es%20OpenCV.. [Último acceso: 06 05 2023].

[16] J. G. G. A. V.M. Arevalo, «Open-Source,» 2023. [En línea]. Available: http://mapir.isa.uma.es/varevalo/drafts/arevalo2004lva1.pdf. [Último acceso: 17 06 2023].

[17] D. E. King, «Dlib c++ Library,» Revista de investigación de aprendizaje automático, 2009. [En línea]. Available: http://dlib.net/ml.html. [Último acceso: 06 05 2023].

[18] «decodigo.com,» 2023. [En línea]. Available: https://decodigo.com/mapeo-facial-con-dlib-y-python. [Último acceso: 06 05 2023].

[19] «pypi.org,» 20 02 2020. [En línea]. Available: https://pypi.org/project/face-recognition/. [Último acceso: 06 mayo 2023].

[20] M. P. Usaola, «MongoDB: gestión, administración y desarrollo de aplicaciones,» Macario Polo Usaola, 2015. [En línea]. Available: https://books.google.es/books?hl=es&lr=&id=UiAvCwAAQBAJ&oi=fnd&pg=PA7&dq=base+de+datos+MongoDB&ots=SkuY9DAHKl&sig=UMq7wUuq41Cvr1ilehvFBC7eQ4g#v=onepage&q=base%20de%20datos%20MongoDB&f=false. [Último acceso: 17 06 2023].

[21] «CCNA,» 2023. [En línea]. Available: https://ccnadesdecero.es/que-es-websocket/. [Último acceso: 18 06 2023].

[22] «ultramsg,» 2023. [En línea]. Available: https://ultramsg.com/es/#whatsappapi. [Último acceso: 2023 06 22].

[23] «WIKIPEDIA,» 28 04 2023. [En línea]. Available: https://es.wikipedia.org/wiki/CodeIgniter. [Último acceso: 2023 06 04].

[24] R. KeepCoding, «KEEPCODING,» 14 09 2022. [En línea]. Available: https://keepcoding.io/blog/que-es-bootstrap-5/#:~:text=Bootstrap%205%20es%20una%20de,con%20una%20plantilla%20de%20inicio.. [Último acceso: 08 06 2023].

[25] «pypi.org,» 20 02 2020. [En línea]. Available: https://pypi.org/project/face-recognition/. [Último acceso: 08 06 2023].

[26] L. A. Bucki, Word 2013 Bible, John Wiley & Sons, 2013.

[27] CFI, «Cursos de Formación en Informática,» [En línea]. Available: http://cursosinformatica.ucm.es. [Último acceso: 01 06 2019].

[28] D. G. d. Tráfico, «DGT,» 22 01 2023. [En línea]. Available: https://www.dgt.es/muevete-con-seguridad/evita-conductas-de-riesgo/Conducir-con-sueno-o-cansancio#:~:text=La%20somnolencia%20interviene%2C%20directa%20o,somnolencia%20es%20un%20factor%20implicado.

[29] Dr.K.S.Tiwari, «Cosmos Impact Factor,» 2019. [En línea]. Available: http://ijrar.com/upload_issue/ijrar_issue_20543712.pdf. [Último acceso: 24 01 2023].

[30] B. Guerra, «Transporte.com,» 2022. [En línea]. Available: https://www.transportexxi.com/iot-y-la-revolucion-sostenible-de-la-logistica/. [Último acceso: 24 01 2023].

[31] C. Pascual, «https://programarfacil.com/,» 2022. [En línea]. Available: https://programarfacil.com/esp32/esp32-cam/. [Último acceso: 09 04 2023].

[32] C. Egea, «EUROPA PRESS,» 14 07 2022. [En línea]. Available: https://www.infosalus.com/salud-investigacion/noticia-falta-sueno-volante-comparable-tener-tasa-alcoholemia-sangre-010-20220714123614.html. [Último acceso: 06 05 2023].

[33] «Wikipedia,» 05 14 2023. [En línea]. Available: https://en.wikipedia.org/wiki/Dlib. [Último acceso: 08 06 2023].

[34] «MES,» 24 07 2022. [En línea]. Available: https://omes-va.com/como-instalar-dlib-en-windows-

python/#:~:text=Dlib%20es%20una%20librer%C3%ADa%2Fbiblioteca,puntos%20clave%20en%20el%20rostro.. [Último acceso: 08 06 2023].

[35] P. R. d. l. Santos, «Telefónica Tech,» 22 09 2022. [En línea]. Available: https://empresas.blogthinkbig.com/breve-historia-de-internet-de-las-cosas-iot/. [Último acceso: 08 06 2023].

[36] «Plain Concepts,» 03 01 2023. [En línea]. Available: https://www.plainconcepts.com/es/iot-transporte/#:~:text=El%20IoT%20en%20el%20transporte,l%C3%ADmites%20de%20kil%C3%B3metros%20por%20hora.. [Último acceso: 08 06 2023].

[37] «IEEEXplore,» 08 2013. [En línea]. Available: https://ieeexplore.ieee.org/abstract/document/6602353. [Último acceso: 08 06 2023].

[38] «MDPI,» 30 10 2013. [En línea]. Available: https://www.mdpi.com/1424-8220/14/1/1106. [Último acceso: 08 06 2023].

[39] A. a. J. P. Everlet, « Introducción al Internet de las cosas.,» 2013. [En línea]. Available: https://cmapspublic2.ihmc.us/rid=1P6R5NKLS-SR8LMD-4NT1/Construyendo_un_proyecto_de_IOT.pdf. [Último acceso: 17 06 2023].

[40] «Astera,» 05 09 2022. [En línea]. Available: https://www.astera.com/es/type/blog/mongodb-vs-mysql/. [Último acceso: 18 06 2023].

[41] «Visual Studio,» 2022. [En línea]. Available: https://visualstudio.microsoft.com/es/downloads/. [Último acceso: 21 06 2023].