

Índice

1. Glosario de Abreviaturas	11
2. Resumen	12
3. Abstract	12
4. Capítulo I: Introducción	13
4.1. Planteamiento del problema	14
4.2. Propuesta de solución	15
4.3. Alcances	16
4.4. Justificación	17
4.5. Objetivos	17
4.5.1. Objetivo General	17
4.5.2. Objetivos Específicos	17
4.6. Metodología	18
4.7. Escenario de pruebas	19
5. Capítulo II: Marco de Referencia	21
5.1. Marco Teórico	21
5.1.1. Somnolencia	21
5.1.2. Jetson Nano	21
5.1.3. Dispositivo Portátil	22
5.1.4. Visión Artificial	22
5.1.5. Aprendizaje Profundo	22
5.1.6. Lenguaje de programación	23
5.1.7. Librerías	23
5.1.8. Aplicación	23
5.1.9. Frontend	23
5.1.10. Backend	23
5.1.11. End point	23
5.1.12. Web hosting	23
5.1.13. <i>Content Delivery Network</i>	23
5.1.14. <i>Cloud Computing</i>	23
5.1.15. Estándares y Protocolos de Comunicación Inalámbrica	24
5.1.16. LTE	24
5.1.17. Teorema de Shannon-Hartley	26
5.1.18. Geolocalización	26
5.2. Estado del Arte	27
5.2.1. <i>Driver Drowsiness Detection Using Machine Learning with Visual Behaviour</i>	27
5.2.2. <i>Detection Of Drowsiness And Distraction Of Drivers Using CNN</i>	27
5.2.3. <i>Driver Drowsiness Detection System Using Convolutional Neural Networks</i>	27
5.2.4. <i>Raspberry Pi based System for Visual Object Detection and Tracking</i>	27

5.2.5. Diseño e implementación de sistema de visión artificial para alerta y detección de somnolencia mediante aprendizaje profundo aplicable en conductores de vehículos	27
5.2.6. Aplicación de visión por computador y machine learning al guiado de un robot móvil basado en Raspberry Pi	28
5.2.7. Evaluación de la plataforma Nvidia Jetson Nano para aplicaciones de visión artificial	28
5.2.8. <i>Comparative Study of Computer Vision Based Line Followers using Raspberry Pi and Jetson Nano</i>	28
5.2.9. Sistema de detección de somnolencia mediante inteligencia artificial en conductores de vehículos para alertar la ocurrencia de accidentes de tránsito	28
5.2.10. Diseño de un sistema electrónico para detectar la somnolencia en automovilistas por medio de la actividad ocular	29
5.2.11. Sistema para la detección del estado de somnolencia en seres humanos, con reconocimiento de patrones	29
5.2.12. Sistema de Detección de Somnolencia	29
5.2.13. Desarrollar un prototipo de reconocimiento facial basado en Machine Learning para detectar estado de Somnolencia en conductores de una cooperativa de transporte	29
5.2.14. Sistema basado en la detección y notificación de somnolencia en conductores de autos	30
5.2.15. Detección de somnolencia para conducción sin accidentes	30
5.2.16. Diseño e implementación de un sistema de geolocalización en interiores para plataforma Android vía la red enterprise WLAN de la PUCP	30
5.2.17. Propuesta de un sistema de geolocalización y monitoreo vía GPS/GSM/GPRS aplicado a un pulsómetro para personas con enfermedades cardiovasculares	31
5.2.18. Influencia de un sistema de geolocalización en el control y monitoreo de vehículos con dispositivos GPS en una empresa logística	31
6. Capítulo III: Análisis	32
6.1. Análisis y delimitación de la zona geográfica	33
6.2. Análisis del Módulo Central de Procesamiento	35
6.2.1. Análisis y Elección del microordenador	37
6.2.2. Análisis y Elección de la alarma	38
6.2.3. Análisis y Elección de la unidad de almacenamiento externa	39
6.2.4. Análisis y elección de lenguajes de programación para el microordenador	40
6.2.5. Análisis del Submódulo Visión Artificial	42
6.2.6. Análisis y Elección de herramientas de programación	48
6.2.7. Análisis y Elección de la cámara digital	48
6.2.8. Análisis y Elección de algoritmos de visión artificial	49
6.3. Análisis del Módulo de Comunicaciones	50
6.3.1. Análisis del Submódulo de Telemetría	51
6.3.2. Análisis del Submódulo de Datos	51
6.3.3. Análisis del Submódulo de Cobertura	52
6.4. Análisis del Módulo de Estación Base	54
6.4.1. Análisis de la Aplicación Web	59

6.4.2. Análisis y Elección del manejador de bases de datos	68
6.4.3. Análisis y Elección de lenguajes de programación web	68
6.4.4. Análisis y Elección del servidor de alojamiento	70
7. Capítulo IV: Diseño	73
7.1. Diseño del Módulo Central de Procesamiento	73
7.1.1. Elección de la Unidad Contenedora de Procesamiento	75
7.1.2. Diseño del Submódulo de Visión Artificial	79
7.2. Diseño del Módulo de Comunicaciones	84
7.3. Diseño de la Estación Base	86
7.3.1. Diseño de la Aplicación Web	88
7.3.2. Diseño de la Base de Datos	95
8. Implementación	98
8.1. Módulo Central de Procesamiento	98
8.1.1. Instalación del entorno de desarrollo en la NVIDIA Jetson Nano	98
8.1.2. Implementar algoritmo para la detección del rostro	103
8.1.3. Implementar puntos de referencia en boca y ojos	105
8.1.4. Implementar metrica MOR y medición de parpadeos	105
8.1.5. Implementar algoritmo para la detección somnolencia	107
8.1.6. Integración de los periféricos y accesorios al Módulo Central de Procesamiento	109
8.2. Módulo de Comunicaciones	110
8.2.1. Investigación de la documentación del módulo 3G/4G LTE-Base Hat	110
8.2.2. Implementación del sistema de Geolocalización	112
8.2.3. Conexión de la aplicación web con el Módulo Central de Procesamiento	118
8.3. Módulo de Estación Base	122
8.3.1. Definición de rutas del frontend	122
8.3.2. Definición de rutas del backend	126
8.3.3. Creación de la base de datos no relacional	129
8.3.4. Conexión backend con la base de datos	131
8.3.5. Sistema de acceso con credenciales	132
8.3.6. Creación de los servicios backend	135
8.3.7. Maquetación web	142
8.3.8. Enlace de Amazon S3 con el sistema backend	145
8.3.9. Despliegue de la aplicación web	148
9. Pruebas	150
9.1. Módulo de Visión Artificial	150
9.1.1. Prueba de detección y clasificación de somnolencia:	150
9.2. Módulo de Comunicaciones	150
9.2.1. Prueba de Conectividad LTE:	150
9.2.2. Prueba de Posicionamiento GPS:	150
9.3. Estación Base (Amazon Amplify)	150
9.3.1. Prueba de Integración con GitHub:	150
9.4. Gestión de Contenido Multimedia y Seguridad	150
9.4.1. Prueba de Almacenamiento en S3:	150
9.4.2. Prueba de Autenticación y Gestión de Identidades (Amazon Cognito):	150

9.5.	Base de Datos (DynamoDB)	151
9.5.1.	Prueba de Rendimiento de DynamoDB:	151
9.6.	Aplicación Web (Amplify y React)	151
9.6.1.	Prueba de Rendimiento del Backend:	151
9.6.2.	Prueba de Interfaz de Usuario (Frontend React):	151
9.7.	Pruebas de Integración del Sistema	151
9.7.1.	Prueba de Interacción entre Módulos:	151
9.7.2.	Prueba de Alertas y Monitoreo en Tiempo Real:	151
9.8.	Evaluación del Sistema Final	151
9.8.1.	Prueba de Funcionalidad Integral:	151
9.8.2.	Prueba de Resiliencia del Sistema:	151
9.8.3.	Prueba de Seguridad Integral:	151
9.8.4.	Prueba de Usabilidad:	152
9.9.	Resultados y Ajustes:	152
10.	Validación	153
10.0.1.	Resultados	153
11.	Conclusion y aportaciones	154
12.	Referencias	156

Índice de figuras

1.	Diagrama general del diseño de la arquitectura del sistema.	16
2.	Modelo espiral del proceso de software. [1].	18
3.	Kit para Desarrolladores Jetson Nano [2].	21
4.	Arquitectura de una CDN [3].	23
5.	Diagrama del sistema dividido en módulos.	32
6.	Mapa de cobertura 3G/4G/5G en México.[4]	34
7.	Mapa de cobertura 3G/4G/5G en la Ciudad de México.[4]	34
8.	Programa clásico vs Machine Learning	40
9.	Diagrama de Casos de Uso - Submódulo de Visión Artificial	44
10.	Modulo de Cámara IMX219-160	48
11.	Esquema <i>handover</i> entre dos celdas	52
12.	Mapa de cobertura de la red LTE	53
13.	Diagrama de Casos de Usos del Módulo de Estación Base	57
14.	Diagrama de Casos de Uso - Aplicación Web	65
15.	Planes de Alojamiento AWS Amplify	71
16.	Tiempos de Respuesta de servidores de HostGator	72
17.	Diagrama de Secuencia - Módulo Central de Procesamiento	74
18.	Dimensiones Nvidia Jetson Nano	75
19.	Nvidia Jetson Nano	76
20.	Zumbador Pasivo KY-006	76
21.	Modulo de Cámara IMX219-160	76
22.	Jumpers	77
23.	Micro SD	77
24.	Carcasa Waveshare	78
25.	Componentes de la Unidad Contenedora de Procesamiento	78
26.	Diagrama de Actividades - Submódulo de Visión Artificial	80
27.	Diagrama de Flujo - Submódulo de Visión Artificial	81
28.	Puntos de referencia	82
29.	<i>Mouth Opening Ratio</i>	82
30.	Diagrama de Flujo del Módulo de Comunicaciones	84
31.	Diagrama Actividades del Módulo de Comunicaciones	85
32.	Diagrama de Diseño - Módulo de Estación Base	86
33.	Diagrama de Comunicación - Módulo de Estación Base	87
34.	Diagrama de Clases - Aplicación Web	88
35.	Diagrama de Secuencia Detalle Reporte Incidencia	89
36.	Diagrama de Secuencia Consultar Ubicación	89
37.	Diagrama de Secuencia Recuperar Contraseña	89
38.	Diagrama de Secuencia Confirmar Incidencia	90
39.	Diagrama de Secuencia Consultar Perfil	90
40.	Diagrama de Secuencia Registrar Conductor	91
41.	Diagrama de Secuencia Modificar Conductor	91
42.	Diagrama de Secuencia Eliminar Conductor	91
43.	Diagrama de Secuencia Estado de los Periféricos	92
44.	Página Inicio de Sesión	92

45.	Página Principal	93
46.	Página Perfil del Conductor	93
47.	Página Detalle de Incidencia	94
48.	Página Ubicación en Tiempo Real	94
49.	Arquitectura Amazon Cognito	97
50.	Instalación del JetPack SDK realizado con BalenaEtcher.	98
51.	Periféricos de entrada para el Kit de desarrollo Jetson Nano.	99
52.	Importación de librerías instaladas con Python.	100
53.	Interfaz después de la instalación de Jetson Stats.	101
54.	verificación del archivo swap credo.	102
55.	Visualización de la instalación exitosa de OpenCV con Cuda en Jtop.	102
56.	Inicialización de cámara y detector.	103
57.	Implementación del detector de rostro.	103
58.	Puntos de referencia facial.	105
59.	Implementación de la metrica MOR y medición de parpadeos.	106
60.	Cálculo de la Apertura de la Boca.	107
61.	Detección de Parpadeo.	108
62.	Detección de Parpadeo.	108
63.	Visualización en Tiempo Real.	109
64.	Visualización de FPS.	109
65.	Carcasa sin armar.	110
66.	Carcasa armada con la Jetson Nano y periféricos integrados.	110
67.	Módulo Base-Hat SIM7600G-H	110
68.	Instalación de librerías	111
69.	Instalación de librerías	111
70.	Minicom	111
71.	Actualización de drivers	112
72.	Establecer dirección IP	112
73.	Intalaci?n de bibliotecas	112
74.	Instalaci?n de comandos	113
75.	Prueba de SIM7600G	113
76.	Comandos GPS para pruebas.	113
77.	C?digo que permite obtener la posición GPS parte 1	113
78.	Código que permite obtener la posición GPS parte 2	114
79.	C?digo que permite oibtener la posici+on GPS parte 3	114
80.	Código que permite obtener la posición GPS parte 4	115
81.	Mapa creado en AWS - MyMap	116
82.	Código para obtener la ubicación geográfica en la aplicación web	117
83.	Coordenadas geográficas en la Jetson Nano	117
84.	Detalles del mapa creado en AWS	118
85.	Coordenadas geográficas que recibe la página web	118
86.	Abrir minicom por comando.	118
87.	Verificar conexión.	119
88.	Descarga del driver.	119
89.	Instalar driver.	119
90.	Comando para comprobar interfaz wwan0.	119

91.	Habilitar la interfaz wwan0	119
92.	Comunicar por Minicom	120
93.	Asignar IP	120
94.	Red 4G LTE activa por medio de la interfaz wwan0	121
95.	Conexión con boto3 hacia AWS	121
96.	Creación proyecto de react	122
97.	Instalación React Router DOM	122
98.	Función RequireAuth	122
99.	Componente Login	123
100.	Ruta protegida del componente Home	123
101.	Directorio de rutas	124
102.	Página de Login	124
103.	Instalación Amplify CLI	126
104.	Configuración del proyecto	126
105.	Implementación del SDK de Amplify	126
106.	Creación de aplicación de Express	126
107.	Amplify add api	127
108.	Configuración de parámetros de GraphQL	127
109.	Definición de Schemas para el manejo de datos	128
110.	Generación de Endpoint de GraphQL	129
111.	Tablas en DynamoDB	130
112.	Consola de AWS	131
113.	Tablas generadas mediante los schemas definidos	131
114.	Funcionamiento de Appsync	132
115.	Generación de endpoint de GraphQL	132
116.	Arquitectura Amazon Cognito	133
117.	Implementación de Amazon Cognito en el proyecto	133
118.	Menu de configuración de Amazon Cognito	133
119.	Implementación completada	134
120.	Grupos de usuario	134
121.	Grupos de usuario	135
122.	Grupos de usuario	135
123.	Función getConductor	136
124.	Función listConductors	136
125.	Función getIncidencia	137
126.	Función listIncidencias	137
127.	Función createConductor	138
128.	Función updateConductor	138
129.	Función deleteConductor	139
130.	Función createIncidencia	139
131.	Función oncreateIncidencia	140
132.	Consola de AWS	140
133.	Funcionamiento de Appsync	141
134.	Crear Conductor	141
135.	Página Principal - Layout.jsx	142
136.	Vista Reporte Incidencia Incidencia - Incidencia.jsx	143

137. Vista Ubicacion	143
138. Vista Conductores - Conductores.jsx	144
139. Configuración de servicio de almacenamiento S3	145
140. Generación de Endpoint de GraphQL	145
141. Generación de Endpoint de GraphQL	145
142. Generación de Endpoint de GraphQL	145
143. Generación de Endpoint de GraphQL	146
144. Generación de Endpoint de GraphQL	146
145. Generación de Endpoint de GraphQL	146
146. Tablas generadas mediante los schemas definidos	147
147. Amplify Hosting	148
148. ramificación de repositorio de Github	149
149. Acceso a la página web desde el navegador	149

Índice de tablas

1.	RF01- Establecer conexión con la base de datos	35
2.	RF02- Activar Alarma	35
3.	RF03- Enviar Video de Incidencia	35
4.	RF04- Realizar Reporte de Incidencia	36
5.	RF05- Enviar Reporte de Incidencia a la Base de Datos	36
6.	RF06- Eliminar Video de Incidencia	37
7.	Comparación de microordenadores	37
8.	Cuadro Comparativo de modelos de MicroSD	39
9.	Tabla Comparativa Lenguajes de Programación	41
10.	RF01- Ingresar al Campo Visual	42
11.	RF02- Capturar Video	42
12.	RF03- Detectar Características Faciales	43
13.	RF04- Clasificar imagen	43
14.	RF05- Enviar señal de Alerta	43
15.	RF06- Almacenar Incidencia	44
16.	Caso de Uso 01 - Ingresar al campo visual de la Cámara	45
17.	Caso de Uso 02 - Capturar Video en Tiempo real	45
18.	Caso de Uso 03 - Detectar Rostro	45
19.	Caso de Uso 04 - Reconocer características faciales	46
20.	Caso de Uso 05 - Detectar Signos de Somnolencia	46
21.	Caso de Uso 06 - Enviar Señal de Alerta	47
22.	Caso de Uso 07 - Identificar Incidencia	47
23.	Categorías LTE	50
24.	RF01- Guardar Incidencia	54
25.	RF02- Guardar Video	55
26.	RF03- Conectar Aplicación Web	55
27.	RF04- Desplegar Aplicación Web	55
28.	RF05- Guardar Credenciales de Usuario Administrador	56
29.	Requerimientos No Funcionales del Módulo de Estación Base	56
30.	Caso de Uso 01 - Guardar Incidencia	57
31.	Caso de Uso 02 - Guardar Video	58
32.	Caso de Uso 03 - Conectar Aplicación Web	58
33.	Caso de Uso 04 - Desplegar Aplicación Web	58
34.	Caso de Uso 05 - Guardar Credenciales de Usuario Administrador	59
35.	RF01- Iniciar Sesión	59
36.	RF02- Mostrar el Historial de Reportes de Incidencia	60
37.	RF03- Visualizar Reporte de Incidencia	60
38.	RF04- Confirmar Incidencia	61
39.	RF05 - Recuperar Contraseña	61
40.	RF06- Mostrar perfil del Conducto	62
41.	RF07- Mostrar ubicación Geográfica	62
42.	RF08- Descartar Incidencia	63
43.	RF09- Registrar Usuario	63
44.	RF10- Mostrar el Historial de Reportes de Incidencia	63

45.	RF11- Mostrar el Historial de Reportes de Incidencia	64
46.	Requerimientos No funcionales - Aplicación Web	64
47.	Caso de Uso 01 - Iniciar Sesión	65
48.	Caso de Uso 02 - Registrar Usuario	66
49.	Caso de Uso 03 - Modificar Usuario	66
50.	Caso de Uso 04 - Eliminar Usuario	66
51.	Caso de Uso 05 - Confirmar Incidencia	67
52.	Caso de Uso 06 - Visualizar Incidencia	67
53.	Caso de Uso 07 - Recuperar Contraseña	67
54.	Caso de Uso 08 - Visualizar Ubicación en Tiempo Real	68
55.	Cuadro Comparativo de los Sistemas de Gestión de Bases de Datos No Relacionales	69
56.	Tipos de Archivo Multimedia	70
57.	Agrupación de Puntos de Referencia	82

1. Glosario de Abreviaturas

Glosario de Abreviaturas

- **CPU** *Central Processing Unit*
- **GPU** *Graphical Processing Unit*
- **LTE** *Long Term Evolution for Machines*
- **RAM** *Random Access Memory*
- **SBGD** Sistema Gestor de Base de Datos
- **SQL** *Structured Query Language*

2. Resumen

El sistema propuesto para la detección de somnolencia en conductores utiliza técnicas de visión artificial para analizar en tiempo real la posición del rostro, los ojos y la boca a través de un flujo de video. Se clasifican los estados de los ojos y la boca para identificar síntomas de somnolencia mediante puntos de referencia faciales. La medición de la relación de apertura en los ojos y la boca se realiza mediante el seguimiento de parpadeos y bostezos. En caso de detectar somnolencia, el sistema activa una alarma y genera un reporte de incidencia enviado a una base de datos. Se emplea una Jetson Nano para la portabilidad del sistema. A su vez, se ha desarrollado una aplicación web que ofrece el monitoreo en tiempo real de la ubicación del conductor, además de proporcionar un sistema de administración que permite a los usuarios acceder y visualizar la información de los reportes de incidencia, así como detalles específicos de cada conductor.

Palabras Clave: Somnolencia, Visión Artificial, Geolocalización, Alerta, Aplicación web, Sistema de Administración.

3. Abstract

The proposed system for detecting drowsiness in drivers uses artificial vision techniques to analyze in real-time the position of the face, eyes, and mouth through a video stream. The states of the eyes and mouth are classified to identify drowsiness symptoms using facial reference points. The measurement of the opening ratio in the eyes and mouth is carried out through tracking blinks and yawns. If drowsiness is detected, the system activates an alarm and generates an incident report sent to a database. A Jetson Nano is used for system portability. Additionally, a web application has been developed that provides real-time monitoring of the driver's location, along with offering an administration system that allows users to access and visualize information from incident reports, as well as specific details for each driver.

Keywords: Drowsiness, Artificial Vision, Geolocation, Alert, Web Application, Management System.

4. Capítulo I: Introducción

De acuerdo con el Informe sobre la situación mundial de la seguridad vial 2018, publicado por la Organización Mundial de la Salud (OMS), se registraron alrededor de 1,35 millones de muertes anuales debido a accidentes de tránsito. Estas cifras alarmantes no solo afectan a los conductores y pasajeros, sino también a los peatones, ciclistas y motociclistas, especialmente en países en desarrollo [5].

Para abordar esta problemática, se busca prevenir la somnolencia al conducir, la cual es reconocida como un factor contribuyente a los accidentes viales. En México, por ejemplo, se estima que anualmente fallecen en promedio 16,500 personas debido a estos percances. Además, los accidentes viales representan un costo significativo para el país, estimado en alrededor de 150 mil millones de pesos, equivalente al 1.7% del Producto Interno Bruto (PIB), considerando costos directos e indirectos [6].

En el desarrollo logístico de empresas que se encargan de transportar a pasajeros o que se encargan de repartir paquetería, los datos pueden verse de forma más alarmante o preocupante, ya que, usualmente, las personas designadas como conductores se enfrentan a largas horas de jornada sin descanso y, en muchos casos, rotan por horarios que pueden ser en el día o en la noche, aumentando el riesgo de accidentarse por el desbalance de tener un horario mixto.

Muchas de estas empresas optan por sistemas de rastreo de vehículos, que hoy en día son un recurso determinante para la planificación de rutas de distribución al optimizarlas y, lo más importante, al estar en constante monitoreo se puede tener la certeza de que se está siguiendo la ruta marcada correctamente [7].

Teniendo en cuenta este contexto, resulta relevante analizar y diseñar un sistema que aborde la problemática de la somnolencia al conducir. Este sistema se basará en el uso de una cámara digital y algoritmos de visión artificial para detectar signos de somnolencia en el conductor. Además, se incluirá un sistema de administración que permitirá la geolocalización en tiempo real, así como la gestión de conductores y reportes de incidencias.

El desarrollo de este sistema busca aprovechar los avances tecnológicos recientes en el campo de la visión artificial y el aprendizaje automático, así como la disponibilidad de microordenadores eficientes y las mejoras en las tecnologías de telecomunicaciones, como el 4G y el Internet de las Cosas (IoT). Estas tecnologías permitirán detectar y alertar los síntomas de somnolencia en los conductores, brindando una solución más asequible en comparación con las opciones tradicionales de rastreo satelital.

En resumen, el objetivo principal del presente trabajo es desarrollar un sistema portátil que pueda detectar la somnolencia al conducir, alertar al conductor y monitorear la ubicación en tiempo real, además de gestionar las incidencias detectadas.

4.1. Planteamiento del problema

La somnolencia es un fenómeno complejo de analizar debido a los factores que pueden intervenir. Algunas de las características más notorias de un estado de somnolencia se pueden apreciar principalmente en el rostro de las personas: frecuencia de parpadeo, bostezos, movimientos faciales y cabeceos, los cuales son parámetros clave para determinar si una persona está en estado de somnolencia o vigilia.

Gracias a los avances tecnológicos en los últimos años, se han desarrollado técnicas de visión artificial y aprendizaje automático que permiten detectar patrones de manera más eficiente. En el área del hardware, se han desarrollado microordenadores capaces de realizar tareas que requieran un nivel moderado de computación de manera eficaz. Finalmente, en el área de telecomunicaciones, tecnologías como el 4G y avances en el *Internet of Things* han permitido mayores velocidades de transmisión de datos, así como mayor cobertura dentro del territorio nacional [8].

Hoy en día, se están comenzando a utilizar tecnologías para prevenir y detectar síntomas de fatiga y somnolencia en conductores.

Uno de estos métodos está basado en el comportamiento del vehículo, el cual detecta el estado del conductor mediante el análisis de distintas métricas como son: movimientos del volante, posición del vehículo, la presión del acelerador o del freno, cambio de velocidades, con los cuales se determina la posibilidad de que el conductor se encuentre en estado de somnolencia. El principal problema de dicho método es que las características individuales del vehículo, del conductor y de la carretera repercuten en la eficacia del sistema.

Por otra parte, se encuentran los métodos que se basan en el análisis de variables fisiológicas, los cuales permiten la detección de somnolencia en sus fases tempranas con una baja tasa de falsos positivos. Se destacan los métodos basados en: electroencefalograma (EEG), electromiograma (EMG), electrocardiograma (ECG) y electrooculograma (EOG). Entre todos los métodos, el EEG es el más común para la detección de la somnolencia, donde se analizan diferentes bandas de frecuencia. Todas estas señales brindan información adicional al momento de analizar el estado de somnolencia de una persona. Sin embargo, estos métodos requieren contacto con el conductor y el uso excesivo de canales de encefalogramas, lo cual resta comodidad, maniobrabilidad y practicidad al conductor, además de poder llegar a ser invasivos, lo cual puede entorpecer el desempeño del conductor.

Finalmente, se encuentra el análisis de características visuales que puede presentar un conductor somnoliento, como los movimientos faciales, parpadeos rápidos y constantes, cabeceos y bostezos frecuentes. Sin embargo, los bostezos se presentan generalmente antes de que el conductor entre en somnolencia, mientras que los cabeceos normalmente ocurren cuando el conductor se duerme. Por lo que estos métodos no son capaces de detectar con exactitud cuando un conductor está empezando a entrar en un estado de somnolencia. Se debe tomar en cuenta las diferencias temporales entre los distintos signos visuales, por lo que realizar la combinación de varias de estas características aumentará la robustez final del sistema, logrando una mejor eficiencia. [9].

Actualmente existen distintas opciones para la geolocalización de vehículos en tiempo real, pero con costos elevados, como lo es el rastreo satelital, el cual requiere un pago por servicio con una

suscripción anual o mensual. Entonces, surge una oportunidad para desarrollar soluciones más asequibles y que sean igualmente eficaces, como es el caso de las redes móviles LTE [10].

Dadas las posibles soluciones antes mencionadas, se plantea la siguiente pregunta: ¿Cómo desarrollar un sistema portátil que pueda detectar la somnolencia, alertar al conductor, además de monitorear la ubicación en tiempo real y gestionar las incidencias detectadas en el conductor?

4.2. Propuesta de solución

Como respuesta a la problemática planteada en la sección 4.1, se propone el desarrollo de un sistema portátil que sea capaz de detectar somnolencia en conductores y a su vez activar una alarma que permita alertar al conductor. Como parte del sistema, un subsistema permitirá la administración y validación de las incidencias reportadas, así como el monitoreo GPS del conductor en tiempo real dentro de la Ciudad de México.

Los problemas que resolverá el sistema se listan a continuación:

- El sistema será portátil, por lo que se hará uso de un microordenador para poder ser instalado y que funcione dentro de un vehículo.
- Mediante una cámara digital conectada al microordenador, el sistema analizará el rostro del conductor en tiempo real, tomando en cuenta como parámetros de análisis los ojos y la boca del conductor.
- El sistema hará uso de técnicas de visión artificial para poder detectar si el conductor presenta signos de somnolencia.
- En caso de presentarse un caso de somnolencia, el sistema activará una alarma para alertar al conductor.
- Al mismo tiempo, el sistema realizará y almacenará un reporte de incidencia que contendrá datos como fecha, hora, ubicación geográfica y un pequeño videoclip que muestre el momento en que el conductor presentó signos de somnolencia.
- Posteriormente, el sistema hará uso de redes de telecomunicaciones móviles para enviar el reporte de incidencia previamente generado hacia una estación base que funcionará mediante una aplicación web, donde un administrador podrá verificar este reporte con el fin de confirmar que se trata de un caso de somnolencia y no un falso positivo.
- A su vez, el administrador podrá consultar la ubicación en tiempo real del conductor desde la misma estación base.
- En la Figura 1 se muestra la propuesta del diagrama general de diseño de la arquitectura del sistema.

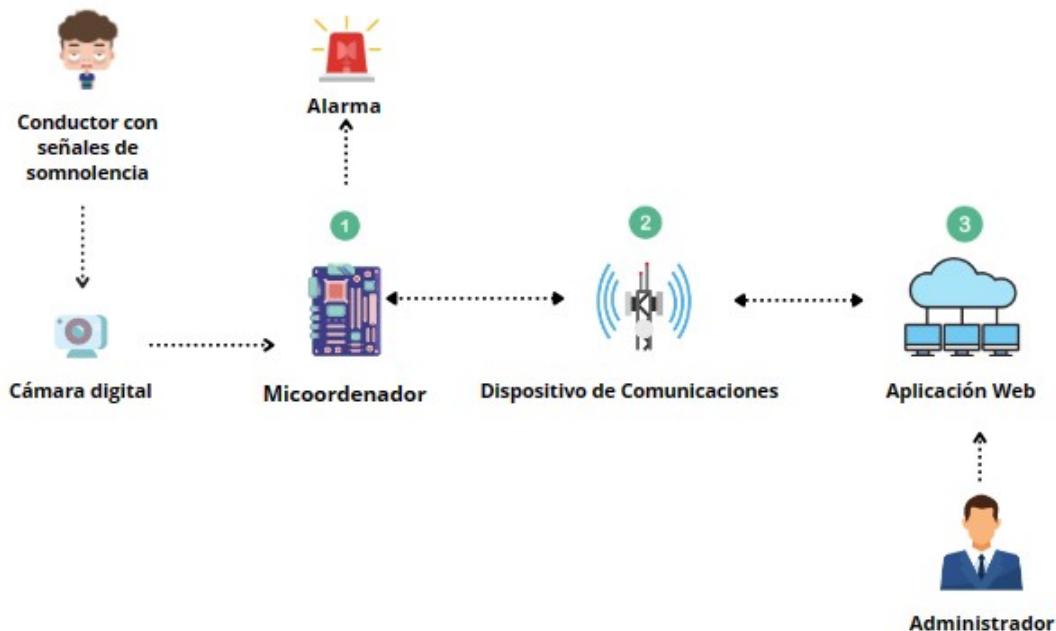


Figura 1: Diagrama general del diseño de la arquitectura del sistema.

4.3. Alcances

A continuación, se describen los alcances de la propuesta de solución:

- Detectar posibles síntomas de somnolencia del conductor.
- Alertar al conductor mediante una alarma en caso de que se detecten síntomas de somnolencia.
- El sistema realizará un reporte de incidencia al detectar somnolencia.
- Solicitar el posicionamiento mediante una tecnología de geolocalización en tiempo real.
- La transmisión de los datos del módulo de procesamiento se realizará mediante una red inalámbrica.
- Se marcará la posición GPS en el mapa del conductor y se visualizará en la aplicación web.
- Las pruebas a realizar se llevarán a cabo en la Ciudad de México, que cuenta con cobertura de redes móviles.
- La aplicación web permitirá consultar las incidencias de cada conductor registrado y la ubicación del conductor en tiempo real.

4.4. Justificación

Actualmente, se han desarrollado diferentes avances tecnológicos tales como el internet de las cosas (IoT) el cual permite la comunicación con dispositivos y da pauta al desarrollo de sistemas inteligentes que resuelven problemas complejos de manera automática y eficaz en ambientes específicos, la inteligencia artificial, el machine learning, el procesamiento masivo de datos en menor tiempo, entre muchos otros.

El sistema propuesto será capaz de detectar la somnolencia apoyado en el análisis de los aspectos fisiológicos del rostro, donde se delimitan los datos biométricos del estado de somnolencia mediante el uso de visión artificial. Estos datos serán posteriormente enviados a una estación base con ayuda de redes inalámbricas.

Al haber accidentes de tránsito por somnolencia en México, este proyecto tiene como objetivo implementar un sistema de detección de somnolencia mediante visión artificial y el uso de una alarma auditiva para que el conductor se mantenga alerta. Además, se busca integrar un sistema que permita el monitoreo del vehículo mediante la geolocalización del mismo y un sistema interactivo con el usuario. Este sistema permitirá visualizar las incidencias que puedan presentarse por parte de los conductores, con el fin de obtener registros que nos permitan realizar estadísticas y recabar información relevante para retroalimentar el rendimiento de cada conductor.

Ofreciendo así un sistema portátil de monitoreo, detección de somnolencia y alerta, que incluye una aplicación web con registros detallados de incidencias y estadísticas para evaluar la eficiencia de los conductores.

4.5. Objetivos

4.5.1. Objetivo General

Diseñar un sistema portátil para la detección de síntomas de somnolencia y alerta del conductor. Además de ser capaz de obtener la ubicación del conductor en tiempo real para ser monitoreada desde una aplicación web que a su vez permita gestionar y visualizar los reportes de incidencias del mismo.

4.5.2. Objetivos Específicos

- Diseñar un sistema de visión artificial que sea capaz de detectar y alertar la somnolencia en conductores.
- Integrar herramientas de geolocalización en tiempo real usando redes de telecomunicaciones móviles.
- Diseñar un sistema de administración para la gestión de usuarios, almacenamiento de datos y consultas de las incidencias de cada conductor.
- Integrar un sistema de comunicaciones que permita la interconexión entre el sistema de visión artificial y el sistema de administración.

4.6. Metodología

Espiral

El artículo de Barry Boehm [1], publicado en 1986, introduce la Metodología Espiral como un enfoque para el desarrollo de software que combina elementos de modelos de desarrollo en cascada y prototipos. La metodología espiral se destaca por su enfoque flexible y adaptativo, permitiendo ajustes a lo largo del ciclo de desarrollo.

La Metodología Espiral se basa en la premisa de que el desarrollo de software es un proceso iterativo y evolutivo. Se propone un modelo en espiral dividido en cuatro cuadrantes: determinación de objetivos, evaluación de riesgos, desarrollo y planificación. Cada iteración aborda estos cuadrantes de manera secuencial.

- En la fase de **determinación de objetivos**, se establecen metas específicas para la iteración.
- La **evaluación de riesgos** implica identificar y gestionar proactivamente los riesgos asociados con los objetivos establecidos.
- La fase de **desarrollo** implica la implementación de funcionalidades, utilizando enfoques como prototipos y desarrollo incremental.
- La **planificación** implica revisar el progreso, ajustar planes según sea necesario y prepararse para la próxima iteración.

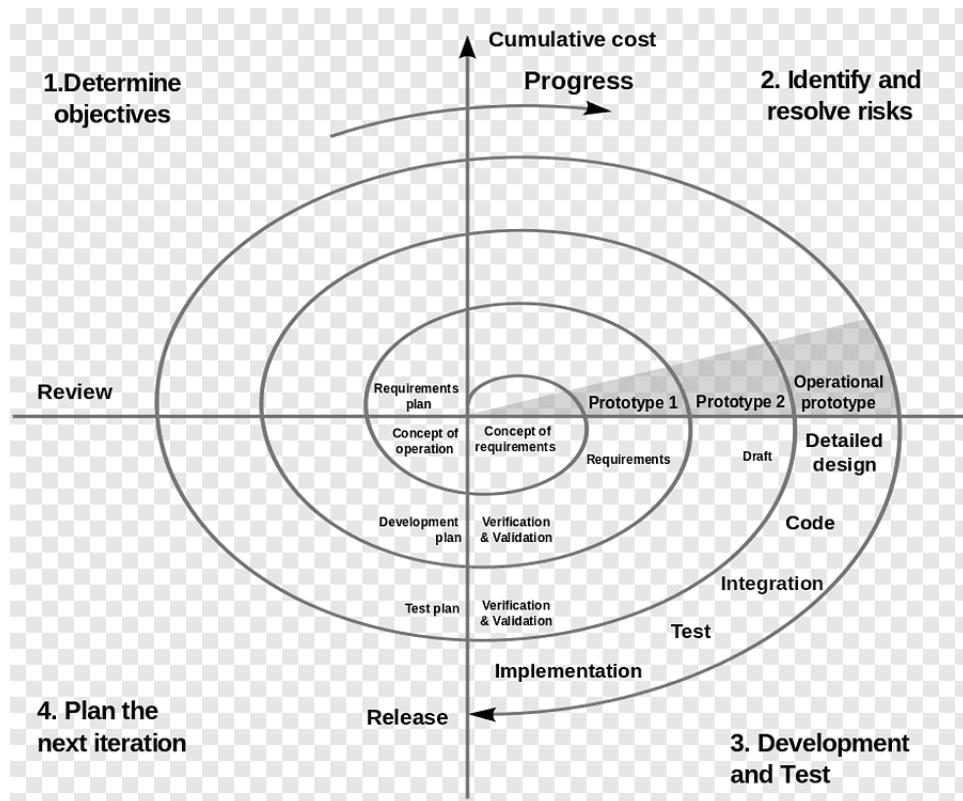


Figura 2: Modelo espiral del proceso de software. [1].

De manera general el proyecto se dividió en tres principales módulos los cuales pasaron por las cuatro fases de la metodología. En la primera fase se especificaron los objetivos particulares de

cada módulo, posteriormente se propuso un prototipo a partir de la elección de los componentes y algoritmos de cada módulo, en la tercera fase se implementaron los modelos propuestos y finalmente en la última fase se revisaron los resultados. A partir de los resultados obtenidos se realizarán los ajustes necesarios para cumplir el objetivo principal y los objetivos específicos. Los puntos anteriores se desglozan a lo largo del siguiente documento.

4.7. Escenario de pruebas

De acuerdo a los objetivos específicos se evaluará.

- El sistema de visión artificial sea capaz de detectar somnolencia.
- Alertar al conductor en caso de detectar incidencia.
- Elaboración de incidencia.
- Conección entre el sistema de visión artificial y el sistema de administración.
- Visualización en tiempo real de la ubicación GPS del conducto usando redes de telecomunicaciones móviles.
- Funcionalidad del sistema de administración para la gestión de usuarios, almacenamiento de datos y consultas de las incidencias de cada conductor.

Las pruebas se dividirán en dos fases:

Fase 1:

En la primera fase se probará el sistema de somnolencia y el sistema de comunicaciones de manera separada, con el propósito de analizar los resultados obtenidos. El sistema de detección de somnolencia será colocado dentro del vehículo, sin embargo, dicho vehículo permanecerá estacionado y con el motor apagado, esto con el fin de realizar pruebas de la precisión del sistema y poder ajustar parámetros en caso de ser necesario.

Estado de los ojos:

- Abiertos, cerrados.
- Duración de estos estados.

Bostezos:

- Apertura de la boca.

Estas pruebas se realizarán durante el día con diferentes conductores para comprobar la detección de somnolencia. Asimismo, se verificará que la alarma se active correctamente al detectar somnolencia en el conductor.

Para el sistema de administración, se probará la aplicación web que incluye el acceso al sistema, la geolocalización del conductor en tiempo real y la visualización de las incidencias.

Fase 2:

En la segunda fase, se probará el sistema completo con el vehículo en movimiento, donde el sistema estará monitoreando el estado de somnolencia del conductor durante el trayecto.

Para el sistema de administración, se corroborará que, en los casos en que el sistema de somnolencia detecte una incidencia, esta será enviada correctamente y podrá ser visualizada desde el sistema de administración. A su vez, se constatará que la información de la ubicación geográfica del sujeto de prueba esté disponible en tiempo real.

5. Capítulo II: Marco de Referencia

5.1. Marco Teórico

5.1.1. Somnolencia

La somnolencia es la necesidad o tendencia que tiene una persona a quedarse dormido. La intensidad de somnolencia está determinada por la calidad de sueño, la cantidad, y el ritmo circadiano de la persona [11].

Se han propuesto tres clases de métodos para poder medir el grado de somnolencia en una persona:

- **Mediciones del comportamiento**

Se basan en la observación del comportamiento del individuo. Entre estos se encuentran: el bostezo, la actividad ocular, expresiones faciales y pestañeo.

- **Test de funcionamiento**

Se usan para medir los efectos de somnolencia en diferentes aspectos del funcionamiento, como por ejemplo: las variaciones en el tiempo de reacción, vigilancia psicomotora y simuladores de manejo.

- **Test Neurofisiológicos**

Son diseñados bajo la premisa de cuantificar la somnolencia de manera objetiva.

5.1.2. Jetson Nano

Jetson Nano es una computadora pequeña y poderosa que permite ejecutar múltiples redes neuronales en paralelo para aplicaciones como clasificación de imágenes, detección de objetos, segmentación y procesamiento de voz. En una plataforma fácil de usar que funciona con 5 vatios [2].



Figura 3: Kit para Desarrolladores Jetson Nano [2].

5.1.3. Dispositivo Portátil

5.1.4. Visión Artificial

La visión artificial constituye un dominio de la inteligencia artificial dedicado a capacitar a las computadoras y sistemas para extraer información valiosa de imágenes digitales, videos y otras fuentes visuales, con el propósito de tomar decisiones o realizar recomendaciones. Este campo depende en gran medida de la disponibilidad de datos abundantes. A través de iterativos análisis de datos, la visión artificial realiza comparaciones repetidas hasta lograr identificar discrepancias y, en última instancia, reconocer patrones visuales en las imágenes. [12].

La visión artificial se refiere a la habilidad de los sistemas computacionales para analizar y obtener información significativa a partir de imágenes y videos. Al emplear técnicas de aprendizaje profundo, las computadoras pueden interpretar las imágenes de manera análoga a la capacidad humana [13]. Este campo de estudio presenta diversas aplicaciones prácticas, algunas de las cuales incluyen:

- Moderación de contenido: eliminación automática de material inseguro o inapropiado en archivos de imágenes y videos.
- Reconocimiento facial: identificación de rostros y reconocimiento de características como la apertura de los ojos, la presencia de gafas y la existencia de vello facial.
- Clasificación de imágenes: identificación de elementos como logotipos de marcas, prendas de vestir, equipo de seguridad y otros detalles visuales en la imagen.

5.1.5. Aprendizaje Profundo

El aprendizaje profundo constituye un enfoque dentro del ámbito de la inteligencia artificial (IA) que busca capacitar a las computadoras para procesar información de manera análoga al cerebro humano. Los modelos de aprendizaje profundo son capaces de reconocer patrones complejos en imágenes, textos, sonidos y otros datos, a fin de generar información y predicciones precisas. Es posible utilizar métodos de aprendizaje profundo para automatizar tareas que habitualmente requieren inteligencia humana, como la descripción de imágenes o la transcripción a texto de un archivo de sonido. Los modelos de aprendizaje profundo son archivos informáticos que los científicos de datos han entrenado para realizar tareas específicas mediante un algoritmo o conjunto predefinido de pasos. Estas herramientas son empleadas por empresas para analizar datos y realizar predicciones en diversas aplicaciones [13].

5.1.6. Lenguaje de programación

5.1.7. Librerías

5.1.8. Aplicación

5.1.9. Frontend

5.1.10. Backend

5.1.11. End point

5.1.12. Web hosting

5.1.13. *Content Delivery Network*

Una Red de Distribución de Contenido, o CDN por sus siglas en inglés. Es una red de servidores interconectados que acelera la carga de las páginas web para las aplicaciones que tienen un uso intensivo de datos. CDN puede significar red de entrega de contenido o red de distribución de contenido. Cuando un usuario visita un sitio web, los datos del servidor de ese sitio tienen que viajar a través de Internet para llegar a la computadora del usuario. Si el usuario se encuentra lejos de ese servidor, un archivo de gran tamaño, como un video o una imagen del sitio web, se demorará mucho en cargar. En su lugar, el contenido del sitio web se almacena en servidores de CDN ubicados geográficamente más cerca de los usuarios, por lo que el contenido llega a sus computadoras mucho más rápido [14].

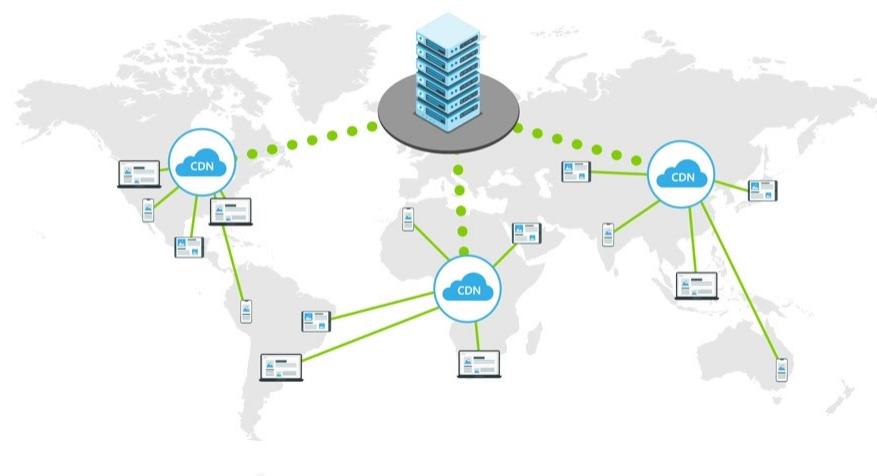


Figura 4: Arquitectura de una CDN [3].

Además de esto, una CDN es capaz de detectar cambios en la información existente, así como detectar la disponibilidad de nuevo contenido en los servidores de origen.

5.1.14. *Cloud Computing*

El Cloud Computing, o computación en la nube, se refiere a la entrega de servicios informáticos a través de Internet, permitiendo el acceso bajo demanda a recursos compartidos como almacenamiento, servidores, bases de datos, redes y software. En lugar de tener que mantener y administrar

infraestructuras locales, los usuarios pueden utilizar recursos en la nube de forma flexible y escalable, pagando solo por lo que utilizan [15].

5.1.15. Estándares y Protocolos de Comunicación Inalámbrica

Para transferir datos o información de un punto a otro sin la utilización de cableado o algún medio físico, existen las redes inalámbricas que utilizan ondas de radio para conectar a los dispositivos permitiendo así, a los dispositivos remotos, se conecten sin dificultad y sin importar que estos dispositivos estén a unos metros o incluso a varios kilómetros de distancia. Se dividen en 4 tipos dependiendo del alcance requerido y se definen por el estándar 802.11 del IEEE que es el organismo de estandarización internacional.

- **Red inalámbrica de área amplia (WWAN)**

Usan ondas de radio pero transmite a uno o varios puntos de acceso inalámbrico donde un usuario inalámbrico puede conectarse a la red, al disponer de un ancho de banda más elevado ofrece una mejor cobertura.

Como ejemplo de estas redes se tienen la tecnologías 4G y 5G. Son conocidas como redes de largo alcance con cobertura de hasta 100km, pueden dar soporte a gran parte del territorio geográfico [16].

En esta red se incluyen:

- **Celulares**

Es conocida como la red de telefonía móvil.

- **LPWAN(*Low Power Wide Area Network*): Red de Área Amplia de Baja Frecuencia**

son redes de área amplia y de baja potencia, es un protocolo de transporte inalámbrico de datos que hoy en día se utiliza como uno de los protocolos básicos para la implementación de IoT. Existen varias implementaciones del protocolo LPWAN, tales como Sigfox,LoRaWAN, NB-IoT y LTE. Hay muchas diferencias entre cada una de ellas en cuanto a los esquemas de modulación, el alcance geográfico, la cantidad de información transmitida y a sus capacidades de encriptación y autenticación[17].

5.1.16. LTE

LTE (Long-Term Evolution) es un estándar inalámbrico de cuarta generación (4G) que proporciona mayor capacidad de red y velocidad para teléfonos móviles y otros dispositivos celulares en comparación con la tecnología de tercera generación (3G). Las capas superiores de LTE se basan en el Protocolo de control de transmisión/Protocolo de Internet, lo que da como resultado una red basada exclusivamente en el Protocolo de Internet, como la de las comunicaciones por cable. LTE admite transmisiones de datos como tráfico mixto de datos, voz, vídeo y mensajería [18].

LTE ofrece a los usuarios varias funciones, incluidas las siguientes:

- **Transmisión de audio y vídeo.** LTE tiene velocidades de descarga y carga más rápidas que 2G y 3G.
- **Conexión a servicios en tiempo real.** Con voz sobre LTE, los usuarios pueden hablar con otras personas sin experimentar retrasos ni fluctuaciones.
- **Velocidades aún más rápidas con LTE-Advanced.** Las velocidades de descarga y carga con LTE-Advanced son dos o tres veces más rápidas que las de LTE estándar. Todos los dispositivos LTE Advanced son compatibles con versiones anteriores del estándar LTE.
- **Agregación de operadores.** Esta función LTE-Advanced mejoró la capacidad de la red, agregando ancho de banda de hasta 100 MHz en cinco operadores (bandas) componentes con un ancho de banda de 20 MHz cada uno. Los teléfonos LTE-A combinan frecuencias de múltiples operadores de componentes para mejorar la señal, la velocidad y la confiabilidad.

5.1.17. Teorema de Shannon-Hartley

Un sistema óptimo es el que cuenta con la capacidad de minimizar la probabilidad de error de bit a la salida del sistema, esto depende de las restricciones de la energía transmitida y del ancho de banda del canal.

El teorema de Shannon-Hartley establece la máxima cantidad de información que puede ser transmitida sin error con un ancho de banda específico y que está expuesto a la interferencia de ruido. La ecuación para la capacidad de canal es:

$$C = B \cdot \log\left(1 + \frac{S}{N}\right) \quad (1)$$

Donde C es la capacidad de canal, es decir, la velocidad máxima a la que se puede transmitir la información a lo largo del canal sin error, medida en bits por segundo, B es el ancho de banda en hertz, S es la potencia de la señal útil en watts y N es la potencia de ruido presente en el canal expresada en watts. Al término S/N se le conoce como relación señal a ruido [19].

5.1.18. Geolocalización

La Geolocalización consiste en la identificación de la posición de un dispositivo móvil en el espacio real. El Sistema de Posicionamiento Global GPS por sus siglas en inglés es la forma más común y precisa en que se realiza la localización geográfica, y es capaz de ubicar el aparato con una precisión de unos pocos metros.

El GPS es una red satelital que cuenta con al menos 30 satélites y que se mantienen en órbita alrededor de la tierra. Si bien el sistema en sus inicios tenía un propósito militar, en la actualidad cualquier persona puede ocuparlo. Cuando se solicita el posicionamiento por medio del GPS este envía señales de radio que permiten localizar a los satélites, el centro de comando transmite la información de la órbita, el tiempo y la posición de los otros satélites en el mismo sistema GPS. Estos satélites envían simultáneamente la información de tiempo y órbita a la tierra y finaliza cuando el dispositivo GPS utiliza la información recibida para determinar su localización la cual se interpreta mayormente en dos conjuntos: la latitud y longitud [20].

5.2. Estado del Arte

5.2.1. *Driver Drowsiness Detection Using Machine Learning with Visual Behaviour*

Este trabajo de investigación propone un sistema de detección de signos de somnolencia en conductores utilizando un modelo de Red Neuronal Convolutinal para detectar la posición de los ojos, y OpenCV junto con Dlib para la detección de la boca y realizar el conteo del número de bostezos por minuto. Para alimentar a la red Neuronal, se utilizaron el conjunto de datos de NTHU-DDD. También fue utilizado el método PERCLOS para obtener el número de parpadeos del sujeto de estudio. Este trabajo concluye que las mayores dificultades a la hora de detección de rostros fueron el uso de gafas oscuras, así como cambios en la iluminación [21].

5.2.2. *Detection Of Drowsiness And Distraction Of Drivers Using CNN*

En este trabajo se implementó el aprendizaje automático y el paquete Keras para construir un modelo de CNN, el cual, clasifica si el conductor se encuentra somnoliento o distraído, el sistema emite un tono de alerta al detectar correctamente la somnolencia, dando al conductor una alerta temprana. Se utilizó el clasificador Open CV Haar-Cascade, un clasificador en cascada basado en características, usando sus funciones integradas, se detectó el rostro y la región de los ojos [22].

5.2.3. *Driver Drowsiness Detection System Using Convolutional Neural Networks*

En este trabajo realizado en la Universidad Anurag se presenta una forma de analizar y anticipar la somnolencia del conductor mediante la aplicación de una red neuronal convolucional sobre la cara del conductor de un marco de secuencia. Se usó un conjunto de datos para dar forma y aprobar el modelo, usando redes convolucionales 3D basadas en modelos de múltiples capas de arquitectura de red neuronal repetitiva para detectar la somnolencia del conductor. Tras una sesión de entrenamiento, se obtuvo una precisión que se acerca al 92 % de aceptación [23].

5.2.4. *Raspberry Pi based System for Visual Object Detection and Tracking*

Este trabajo aborda la implementación de un sistema capaz de realizar el seguimiento de varios objetos mediante una cámara digital conectada a una Raspberry Pi. El motor de este sistema fue desarrollado utilizando C++ y se ejecuta sobre un sistema operativo basado en GNU/Linux. Este sistema también logra transmitir las coordenadas y el tamaño de los objetos detectados a otras computadoras dentro de una red de local de Internet [24].

5.2.5. *Diseño e implementación de sistema de visión artificial para alerta y detección de somnolencia mediante aprendizaje profundo aplicable en conductores de vehículos*

En este trabajo realizado en la facultad de Ingeniería de la Universidad Nacional de Trujillo, se desarrolló un sistema de extracción de características faciales tales como pestañas, cejas y bostezos. Para la extracción de regiones de interés se utilizaron cascadas Haar, y la clasificación de estas características se realizó utilizando un modelo de red LeNet. Además el trabajo incluye la creación de una base de datos de las regiones de interés de la cara utilizando imágenes propias

y también utilizando conjunto de datos externos. Para el desarrollo de este trabajo se utilizó el lenguaje de programación Python, junto con las librerías de OpenCV, Tensorflow y Keras [25].

5.2.6. Aplicación de visión por computador y machine learning al guiado de un robot móvil basado en Raspberry Pi

Esta investigación realizada en la Universidad de Sevilla, detalla el desarrollo de un vehículo con guiado autónomo basado en visión artificial. Se utilizó una red neuronal convolucional para realizar el seguimiento de objetos. El sistema desarrollado consta de una integración de una computadora remota a la par de una Raspberry Pi [26].

5.2.7. Evaluación de la plataforma Nvidia Jetson Nano para aplicaciones de visión artificial

Este trabajo de investigación aborda distintos proyectos tales como: Estimación de pose humana en tiempo real y Reconocimiento de matrícula en tiempo real sobre una Nvidia Jetson Nano. Se realizaron pruebas de rendimiento para conocer las limitaciones de la Jetson Nano. Los resultados presentados muestra que la Jetson Nano permite trabajar tanto como imágenes como con vídeos de forma fluida, aunque con ciertas limitaciones en el uso de redes neuronales profundas. Sin embargo, al utilizar la aceleración de hardware ofrecida por Nvidia, se redujo considerablemente los tiempos de ejecución de los procesos [27].

5.2.8. Comparative Study of Computer Vision Based Line Followers using Raspberry Pi and Jetson Nano

Esta investigación realiza una comparación del rendimiento entre la Raspberry Pi y la Jetson Nano aplicado a un sistema de guía utilizando líneas rectas en el suelo. Los resultados de este trabajo muestran que ambas ofrecen un buen rendimiento. Sin embargo, la Jetson Nano demostró ser 20 segundos más rápida en cuanto al procesamiento de imágenes, y obtuvo un porcentaje del 100 % de efectividad al recorrer el trayecto trazado por el circuito de líneas. Por otro lado, la Raspberry Pi, obtuvo un porcentaje de 98 % de efectividad en cuanto al seguimiento del circuito. Por lo anterior, el estudio concluyó que la Jetson Nano ofrece un mejor rendimiento y efectividad en tareas de visión artificial [28].

5.2.9. Sistema de detección de somnolencia mediante inteligencia artificial en conductores de vehículos para alertar la ocurrencia de accidentes de tránsito

Este proyecto fue realizado por estudiantes de la Escuela Profesional de Ingeniería de Perú y consistió en llevar a cabo un sistema para la detección de la somnolencia y la distracción del conductor. El sistema se desarrolló utilizando C# con EmguCV para detectar la distracción y orientación de los ojos utilizando técnicas de visión artificial. Además, cuenta con un sistema de alarma compuesto por un zumbador de 12v, que se activa a recibir la orden el microcontrolador al procesar el sistema de visión artificial junto a una red neuronal [29].

5.2.10. Diseño de un sistema electrónico para detectar la somnolencia en automovilistas por medio de la actividad ocular

Este trabajo realizado por alumnos de la ESIME Culhuacán en el año 2019 detalla el diseño y la implementación de un sistema que se basa en la fusión de dos señales. Una de ellas proviene de la detección del estado de los ojos utilizando información proveniente de una cámara digital. Para lo anterior, se realiza una segmentación de las regiones de la piel y posteriormente se obtiene la ubicación y el rastero de los ojos. La segunda señal se obtiene a partir de los datos proveniente de un acelerómetro colocado sobre la cabeza del conductor, cuya función es detectar los cabeceos asociados con somnolencia. El procesamiento de dicha señal del acelerómetro se lleva a cabo con la ayuda de la Transformada Wavelet Discreta. Estas dos señales son correlacionadas para tener como salida dos alarmas secuenciales que son percibidas por el conductor. La primera le alerta sobre un primer estado de posible somnolencia y la segunda acciona un control difuso para el control momentáneo del auto y la corrección del volante para el seguimiento del carril. Los resultados obtenidos por este trabajo demuestran una eficacia para la detección de ojos cerrados del 86 % y para la detección de cabeceo superior al 90 % [30].

5.2.11. Sistema para la detección del estado de somnolencia en seres humanos, con reconocimiento de patrones

En este artículo se muestra la implementación de un sistema de detección del estado de somnolencia en seres humanos, a través de la identificación de patrones faciales y la frecuencia de parpadeo de los ojos. Utilizando técnicas de inteligencia artificial, visión por computadora y un sistema embebido con cámara integrada para la adquisición de imágenes. El cual permite detectar en tiempo real el estado de fatiga de un conductor automovilístico y su grado de somnolencia, todo con el objetivo de disminuir la tasa de accidentes viales causados precisamente por la somnolencia en México. Se hizo uso del lenguaje de programación Python, bibliotecas como OpenCV, Dlib y Scipy, las cuales, fueron requeridas debido a los modelos predefinidos que establecen una mayor precisión en la detección de puntos faciales específicos, utilizando como referencia el método de predicción de 68 puntos específicos del rostro. El sistema propuesto tiene la característica de funcionar con luz de día en una primera etapa, y la idea es poder implementarlo en cualquier tipo de vehículo automotriz a un costo accesible a la mayoría de los propietarios de vehículos automotrices [31].

5.2.12. Sistema de Detección de Somnolencia

En este trabajo de fin de grado, realizado por un alumno de la Universidad de La Laguna en el año 2022 se estudió el uso de los modelos de Redes Neuronales para la clasificación de imágenes. Enfocado en la resolución del problema de la somnolencia en los conductores. En el cual se utilizó el lenguaje de programación Python, junto con diversas librerías que facilitan la integración del modelo, y otras que ayudan en la captura de las imágenes [32].

5.2.13. Desarrollar un prototipo de reconocimiento facial basado en Machine Learning para detectar estado de Somnolencia en conductores de una cooperativa de transporte

En este trabajo, realizado por alumnos de la Universidad de GUAYAQUIL en el año 2020-2021 se plantea el desarrollo un prototipo para la detección de la somnolencia del conductor de una cooperativa de transporte el cual dicho conductor se había sobrepasado el límite de horas de trabajo,

usando la técnica de reconocimiento facial, basándose específicamente en el estado de los ojos. Para ello se realizó una investigación bibliográfica relacionada con patrones biométricos, inteligencia artificial y programación mediante Machine Learning, así como, las principales variables que permiten identificar un estado de somnolencia. Dentro de esta investigación también se determinan cuáles son los algoritmos a utilizar (CV2, Imutils, etc.) siendo la herramienta Python y su librería principal Visión por Computadora las seleccionadas para el estudio [33].

5.2.14. Sistema basado en la detección y notificación de somnolencia en conductores de autos

En este trabajo, realizado por alumnos de la Universidad de Córdoba en el año 2015 se plantea el desarrollo que permite alertar a los conductores en estado de somnolencia leve, utilizando la tecnología de reconocimiento de objetos de Kinect y la librería OpenCV con el lenguaje C# para el reconocimiento de imágenes. Además, se diseña una aplicación móvil del sistema operativo Android para la notificación de somnolencia utilizando una conexión socket tipo TCP. Para el procesamiento de imágenes, se utilizó el algoritmo Viola-Jones, el cual se basa en una nueva forma de representación de imágenes llamada Integral Image, permitiendo que las características del detector se comunten rápidamente [34].

5.2.15. Detección de somnolencia para conducción sin accidentes

En este trabajo de fin de grado, realizado por un alumno de la Universidad de Alicante en el año 2022 se propuso crear una aplicación real para detectar la somnolencia al volante haciendo énfasis a un bajo coste económico. Donde se realizó una comparativa entre una solución mediante Machine Learning (ML) y Principal Component Analysis (PCA) [35].

5.2.16. Diseño e implementación de un sistema de geolocalización en interiores para plataforma Android vía la red enterprise WLAN de la PUCP

En este proyecto de titulación de la Pontifica Universidad Católica Del Perú del 2016 se desarrolló una aplicación móvil capaz de geolocalizar a un usuario dentro de las instalaciones de la universidad usando la técnica de Huellas de Señal (fingerprinting), que minimiza el error debido a reflexiones y obstáculos, basada en el estimador de máxima verosimilitud (ML por sus siglas en inglés Maximum Likelihood) junto a las mediciones de señal de los Access Points cercanos usando la red Wi-Fi. La tecnología de radiofrecuencia que se usó fue la de redes inalámbricas de área local, Wi-Fi. Que ofrece conectividad por radiofrecuencia, con alcance local a un dispositivo que envíe datos Ethernet desde la ubicación del mismo hasta una conexión a la red fija, que en este caso la universidad contaba con 32 access points que recibirían la señal de datos a través de cobre o fibra. La técnica del fingerprinting, que está dirigida a geolocalización en interiores, consiste en un mapeo de datos que se encuentran en un escenario para luego asociarlos a una localización y almacenarlos en una base de datos, para estimar la localización más probable se utilizó el algoritmo de ML basado en el teorema de Bayes de probabilidad. En las conclusiones señalan que este sistema obtuvo una precisión del 100% en la estimación del ambiente con un error menor de 2.4m en las pruebas realizadas [36].

5.2.17. Propuesta de un sistema de geolocalización y monitoreo vía GPS/GSM/GPRS aplicado a un pulsómetro para personas con enfermedades cardiovasculares

Tesis del Instituto Politécnico Nacional de la Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Zacatenco del año 2018, donde se empleó un sensor de pulso cardiaco el cual proporciona información en tiempo real de los latidos del corazón mientras que un microcontrolador procesa los datos y, en caso de que se obtengan los datos de que se está presentando una taquicardia, el microcontrolador solicita la ubicación al módulo GPS y envía un mensaje de texto a través de GSM/GPRS a la persona designada [37].

5.2.18. Influencia de un sistema de geolocalización en el control y monitoreo de vehículos con dispositivos GPS en una empresa logística

Tesis del año 2015 de la Universidad César Vallejo de Perú donde se investiga de manera profunda las características y detalles de la tecnología GPS para determinar la influencia de un sistema de geolocalización en el control y monitoreo de vehículos con esta tecnología en una empresa logística [38].

Proyecto	Detección de somnolencia	Portátil	Monitoreo GPS en tiempo real	Sistema de administración
Driver Drowsiness Detection Using Machine Learning with Visual Behaviour	sí	No	No	No

6. Capítulo III: Análisis

Partiendo de la propuesta de solución, se dividió el sistema en tres módulos principales: el Módulo de Comunicaciones, el Módulo Central de Procesamiento y el Módulo de la Estación Base, figura 5.

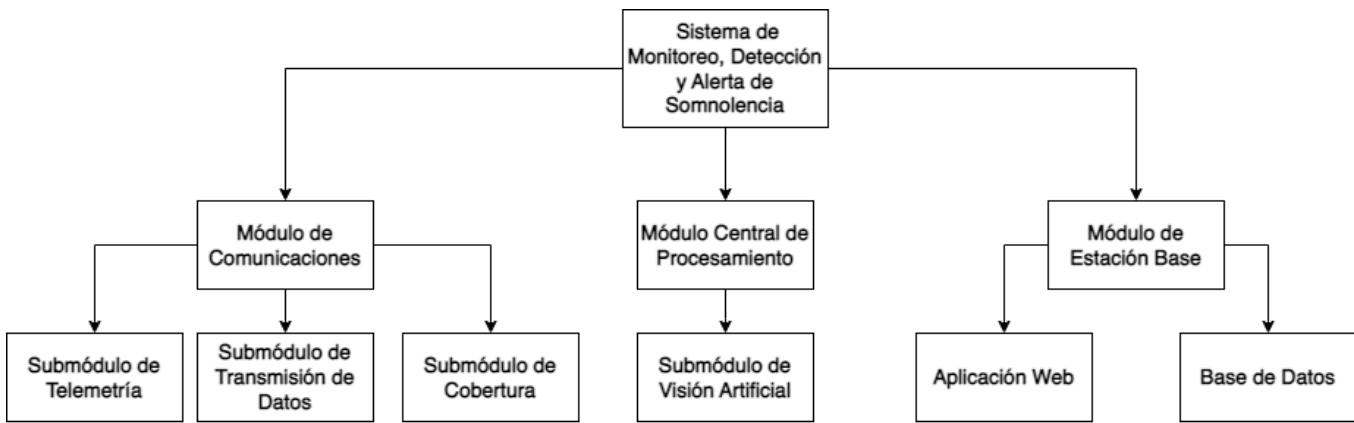


Figura 5: Diagrama del sistema dividido en módulos.

A continuación, se desglosan cada uno de los módulos:

Módulo Central de Procesamiento:

El módulo Central de procesamiento estará compuesto por un microordenador que será colocado dentro de un automóvil. Este microordenador será alimentado por la corriente proporcionada por dicho automóvil. Además, contará con una cámara digital que analizará el rostro del conductor después de que este encienda el automóvil. A este microordenador también le será acoplado un dispositivo que sea capaz de utilizar redes de telecomunicaciones móviles, esto para que sea posible la conexión con la Estación Base. Finalmente, también contará con una alarma para alertar al conductor si este presenta un estado de somnolencia.

Módulo de Visión Artificial

Este submódulo se hará cargo de analizar el rostro del conductor utilizando técnicas de visión artificial. Se tomarán en cuenta los siguientes parámetros para determinar si el conductor presenta o no signos de somnolencia:

- estado de los ojos (cerrados, abiertos).
- estado de la boca (cerrada o abierta).

Si el conductor presenta un estado de ojos cerrados por un tiempo prolongado, se considerará como un estado de somnolencia, por lo que se activará una alarma para alertar al conductor. Así mismo se evaluará el grado de apertura de la boca para determinar si el conductor bosteza y el promedio de bostezos por minuto, si el conductor presenta más de un bostezo dentro de un minuto el sistema

lo clasificará como somnolencia. A su vez el sistema tomará un breve videoclip dónde se aprecie el momento en que el conductor presentó un estado de somnolencia.

Módulo de Comunicaciones

Una vez que se tengan los datos recibidos por el módulo de procesamiento, este módulo se encargará de la transmisión de los datos hacia el módulo de estación base por medio de la tecnología 4G/LTE. Donde la información recibida por el módulo de procesamiento se enviará por medio de la red celular y será subida a la base de datos para después ser descargadas o consultadas por el módulo de Estación Base. La tasa de datos con la que el transceptor funcionará se especifica en la sección [6.3.2](#). En caso de no contar con cobertura, quedarán almacenadas las incidencias en la unidad externa hasta que puedan ser almacenadas una vez que esta se restablezca la comunicación.

Módulo de Estación Base

Este módulo estará compuesto por una aplicación web y una base de datos que se encargará de almacenar las incidencias de los conductores para ser corroborados por un administrador posteriormente, esto con la intención de descartar un falso positivo y realizar los ajustes necesarios al sistema. La aplicación web brindará una base de datos con registros de incidencias por conductor y estadística semanal de incidencias que se presenten durante dicho periodo. Así mismo se podrá consultar la posición gps del conductor.

6.1. Análisis y delimitación de la zona geográfica

El sistema está dirigido principalmente para empresas cuya actividad esté enfocada al transporte de material o personas dentro de la Ciudad de México. Debido a que una parte fundamental del proyecto es la portabilidad, el sistema requiere de un servicio de acceso al internet. Por tanto, se decidió utilizar redes móviles celulares para cumplir con dicho requerimiento.



Figura 6: Mapa de cobertura 3G/4G/5G en México.[4]

Tomando en cuenta la información del mapa de cobertura 3G/4G/5G, de redes móviles en México, mostrado en la figura 6, se observa que las zonas con mayor cobertura son principalmente las que cuentan con mayor densidad de población, por tanto, se plantea que su funcionamiento sea principalmente en la Ciudad de México, debido a que cuenta con mayor infraestructura en dichas redes.

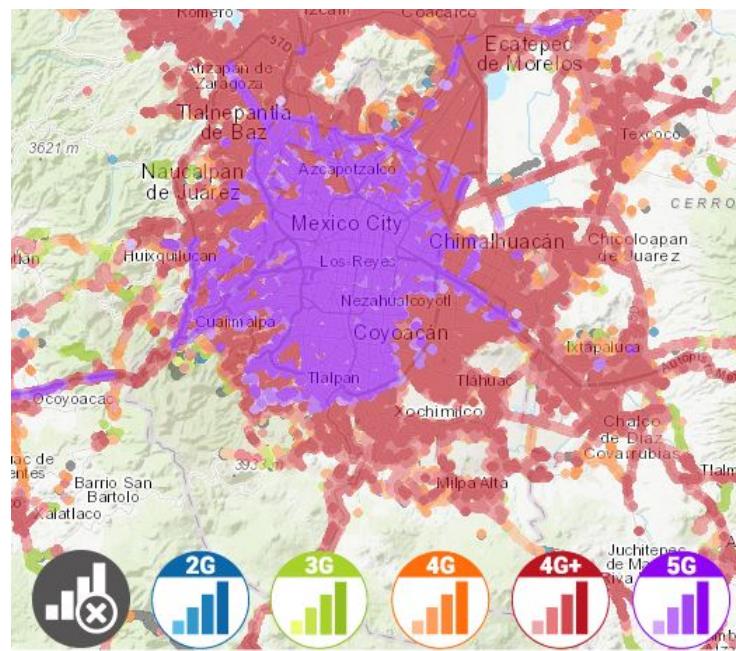


Figura 7: Mapa de cobertura 3G/4G/5G en la Ciudad de México.[4]

6.2. Análisis del Módulo Central de Procesamiento

Análisis de Requerimientos Funcionales

Tabla 1: RF01- Establecer conexión con la base de datos

ID	Nombre corto del Requerimiento
RF01	<i>Establecer conexión con la base de datos</i>
Descripción: Se realizará conexión con la Base de Datos.	
Elementos:	
<ul style="list-style-type: none"> • Microordenador • Base de Datos 	
Solución del Requerimiento:	
Se podrá acceder a la información almacenada y registrar nuevos datos en la base de datos.	

Tabla 2: RF02- Activar Alarma

ID	Nombre corto del Requerimiento
RF02	<i>Activar Alarma</i>
Descripción: El sistema activará el zumbador en caso de que el Submódulo de Visión Artificial detecte somnolencia en el conductor.	
Elementos:	
<ul style="list-style-type: none"> • Microordenador • Alarma 	
Solución del Requerimiento:	
Utilizando métodos de comprobación ofrecidos por cada uno de las librerías de los periféricos, se comprobará que funcionen de manera correcta, en caso contrario se enviará una alerta al módulo de Estación Base.	

Tabla 3: RF03- Enviar Video de Incidencia

ID	Nombre corto del Requerimiento
RF03	<i>Enviar Video de Incidencia</i>
Descripción: Se enviará el video de la incidencia hacia un servicio de almacenamiento en la nube.	
Elementos:	
<ul style="list-style-type: none"> • Video 	
Solución del Requerimiento:	
El video será almacenado y se obtendrá un ID del video.	

Tabla 4: RF04- Realizar Reporte de Incidencia

ID	Nombre corto del Requerimiento
RF04	<i>Realizar Reporte de Incidencia</i>
Descripción: Se realizará una consulta previa a la Base de Datos para obtener el nombre y apellidos del conductor de acuerdo al id del conductor. Se registrará la fecha, hora y ubicación geográfica del momento en que se detectó la somnolencia. Los datos recabados serán usados en la creación de un documento que contenga los datos de la incidencia.	
Elementos:	
<ul style="list-style-type: none"> • Id del Conductor • Nombre de Conductor • Apellidos de Conductor • Fecha • Hora • Ubicación • Id Video 	
Solución del Requerimiento:	
El documento de la incidencia contendrá los campos necesarios para registrar la incidencia en la base d datos.	

Tabla 5: RF05- Enviar Reporte de Incidencia a la Base de Datos

ID	Nombre corto del Requerimiento
RF05	<i>Enviar Reporte de Incidencia a la Base de Datos</i>
Descripción: Se enviará el reporte de incidencia hacia la Base de Datos.	
Elementos:	
<ul style="list-style-type: none"> • Reporte de Incidencia 	
Solución del Requerimiento:	
El Reporte de Incidencia será almacenado en la Base de Datos.	

Tabla 6: RF06- Eliminar Video de Incidencia

ID	Nombre corto del Requerimiento
RF06	<i>Eliminar Video de Incidencia</i>
	<p>Descripción: Se eliminará el video de Incidencia del almacenamiento externo después de ser almacenado en la nube.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Reporte de Incidencia • URL del Video <p>Solución del Requerimiento:</p> <p>El video de la incidencia será eliminado.</p>

6.2.1. Análisis y Elección del microordenador

Como ya se mencionó en la sección 4.2, se hará uso de un microordenador para permitir que el sistema sea portátil y pueda funcionar dentro de un automóvil. Por tanto, se realizó una investigación de los modelos disponibles enfocados a proyectos de visión artificial, figura 7.

Tabla 7: Comparación de microordenadores

Nombre	CPU	RAM	GPU	Alimentación	Precio
Google Coral Dev Board	NXPiMX 8M SoC Dual Core 1.2 GHz	2GB DDR3	Hexa Core A-72 2GHz	5V DC	\$9439
Raspberry Pi 4B	1.5 GHZ 64 bit QuadCore Cortex A-72	1GB, 2GB, 4GB, 8GB DDR4	No cuenta con GPU	5V vía USB-C o puerto GPIO	\$1700
Asus TinkerBoard 2S	Mali-G52 MP6 con 6 núcleos	2GB/4GB dual-channel LPDDR4	Integrated GC7000 Lite Graphics	12V-19V	\$4100
NVIDIA Jetson Nano	ARM A57 2.1 GHz de cuatro núcleos	4 GB de LPDDR4 de 64 bits	NVIDIA Maxwell de 128 núcleos CUDA	5V/4A	\$3,699

Los parámetros que se tomaron en cuenta para la elección de que microordenador utilizar fueron los siguientes:

- Precio
- Voltaje necesario
- RAM
- Módulos compatibles

Si bien la Raspberry Pi es uno de los microordenadores con mayor aceptación por parte de la comunidad, esta no cuenta con una GPU, lo cual reduce significativamente su rendimiento en tareas relacionadas con IA que requieren mayor procesamiento. Por lo tanto, al buscar alternativas

de microordenadores que cuenten con una GPU y basándose en el estudio [28], se concluyó que la Nvidia Jetson Nano ofrece una ventaja superior en comparación con la Raspberry Pi en cuanto a procesamiento y eficacia en tareas que involucran visión artificial. A pesar de que Asus y Google Coral Dev Board cuentan con una GPU, su potencia no supera a la ofrecida por Jetson Nano [39] y su precio es considerablemente mayor, como se puede apreciar en la tabla 7, por lo cual se descartaron como opciones. Debido a lo anterior, se concluyó que la Jetson Nano representa la mejor opción para el desarrollo del presente proyecto debido a su bajo costo, bajo consumo de energía y mejor rendimiento en comparación con sus alternativas.

6.2.2. Análisis y Elección de la alarma

Dentro de la búsqueda de modelos de alarma se encontraron dos tipos de zumbadores, el zumbador activo y zumbador pasivo, donde el primero produce un tono audible fijo, al aplicar una tensión de corriente directa. Mientras que el segundo requiere de una señal oscilante, generalmente de tipo PWM, que indique la frecuencia y la duración de la señal.

Para la elección de la alarma se obtuvieron dos de los principales modelos comerciales de cada tipo de zumbador mencionado:

Buzzer Pasivo KY-006

- Voltaje: 1.5 V - 5 V
- Rango de frecuencia: 1.5 Hz - 2.5kHz
- Decibeles: 72 DB
- Precio: \$50

Buzzer Activo KY-012

- Voltaje: 3.5 V - 5.5 V
- Rango de frecuencia: 300 Hz - 2.5kHz
- Decibeles: 85 DB
- Precio: \$71

Debido a la posibilidad que ofrece el zumbador pasivo para controlar el tono y la duración del mismo, además de su bajo coste en comparación con el zumbador activo KY-012, se optó por el zumbador de tipo pasivo, específicamente el modelo KY-006, en el cual se pueden configurar los tonos desde la Nvidia Jetson Nano por medio del lenguaje Python.

6.2.3. Análisis y Elección de la unidad de almacenamiento externa

Para utilizar una NVIDIA Jetson Nano, se requiere una tarjeta SD que contenga el sistema operativo y las aplicaciones. La tarjeta SD debe tener una capacidad de almacenamiento de al menos 8 GB.

Tabla 8: Cuadro Comparativo de modelos de MicroSD

Micro SD	Almacenamiento	Speed class	Video class	Velocidad de escritura	Velocidad de Lectura	Precio
SanDisk Extreme A2	32-256 (MAX 1T)GB	Clase 30/U3	V30	90 MB/s	160 MB/s	\$1041
Kingston ENDURE-RANCE	32-256 (MAX 1T)GB	Clase 10/U1	-	45 MB/s	95 MB/s	\$628
Samsung EVO Plus	256 GB	Clase 10/U3	V30	130 MB/s	160 MB/s	\$592
Kingston Canvas React Plus V90 Card	64-256 GB	Clase 10/U3	V90	90 MB/s	180 MB/s	\$3596

Tomando en cuenta la capacidad mínima de almacenamiento y el costo de la misma, se decidió usar una unidad de almacenamiento Samsung Evo Plus de 256 GB para la instalación del sistema operativo en la Jetson Nano. Esta elección se fundamenta en las destacadas características de la tarjeta, como su excelente velocidad de escritura, competitiva velocidad de lectura, clasificación de velocidad adecuada y una buena relación entre capacidad y precio.

6.2.4. Análisis y elección de lenguajes de programación para el microordenador

Lo que se debe de considerar para la elección de un lenguaje de programación, es que la programación orientada a la inteligencia artificial es diferente al paradigma de la programación convencional. En esta última, el usuario le indica a la máquina exactamente lo que tiene que hacer, mientras que en Machine learning, se le enseña a programarse sola. Lo cual se ejemplifica en el siguiente gráfico:

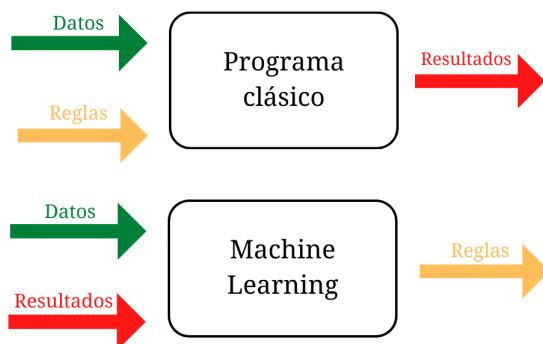


Figura 8: Programa clásico vs Machine Learning

El proceso de trabajo para aprendizaje automático es muy diferente a la construcción de una aplicación convencional. Por este motivo, la manera de utilizar los lenguajes de programación es diferente. Se deben de tomar en cuenta las características que estos utilizan, así como sus enfoques y paradigmas.

Uno de los factores importantes a considerar al momento de elegir un lenguaje de programación orientado a machine learning, es la popularidad el mismo, ya que esta es una señal de la aceptación por parte de la comunidad. A su vez, su soporte es tanto o mas importante, ya que podemos darnos una idea si dicho lenguaje posee las herramientas adecuadas que se acoplan a nuestras necesidades. La velocidad de ejecución es otro factor importante, sobre todo cuando se requiere una minusociudad en la ejecución de procesos y el cuidado de la memoria .Finalmente, la versatilidad del lenguaje es otro factor relevante, ya que, si el lenguaje fue diseñado con una determinada tarea o propósito en mente, este será mucho más eficiente y productivo.

A continuación, se listan algunos de los lenguajes de programación más populares en el campo de Machine Learning:

- Python Es uno de los lenguajes más populares en la comunidad de aprendizaje automático y visión artificial debido a su facilidad de uso, gran cantidad de bibliotecas y frameworks disponibles y gran comunidad de desarrolladores.
- C++: Es un lenguaje de programación de alto rendimiento, es muy adecuado para proyectos de visión artificial que requieren de un gran rendimiento.

- Matlab: Es un lenguaje de programación y entorno de desarrollo específicamente diseñado para matemáticas y cálculo científico, es muy utilizado en el campo de la visión artificial y procesamiento de imágenes.
- Java: Es un lenguaje de programación multiplataforma, es muy popular en el desarrollo de aplicaciones y se utiliza en proyectos de visión artificial.
- R: Es un lenguaje de programación especializado en estadística y análisis de datos. Es ampliamente utilizado en investigación académica, ciencia de datos y análisis estadístico en ciencias sociales, económicas, financieras y biotecnología entre otros.

A continuación se presenta una tabla comparativa de las características que se tomaron en cuenta para la elección sobre qué lenguaje utilizar.

Tabla 9: Tabla Comparativa Lenguajes de Programación

Lenguaje	Popularidad	Versatilidad	Velocidad	Soporte
Python	Alta	Alta	Moderada	Alta
C++	Baja	Alta	Alta	Moderado
R	Alta	Moderada	Moderado	Bajo
Matlab	Baja	Moderada	Moderada	Bajo
Java	Moderada	Moderada	Baja	Moderado

Debido a que en el presente proyecto se realizará la integración de sistemas enfocados a Machine Learning y a su vez a la programación orientada a objetos, Python representa la mejor opción para ser implementado, ya que este lenguaje permite desarrollar ambos ámbitos de una manera integral [40]. Por lo cual será utilizado a lo largo del proyecto.

6.2.5. Análisis del Submódulo Visión Artificial

Submódulo de Visión Artificial:

El submódulo de visión artificial tiene como objetivo realizar la detección de somnolencia del conductor, analizando el rostro con algoritmos de visión artificial.

■ Requerimientos Funcionales

Tabla 10: RF01- Ingresar al Campo Visual

ID	Nombre corto del Requerimiento
RF01	<i>Ingresar al campo visual</i>
	Descripción: El conductor ingresará al campo visual de la cámara digital, posicionándose en el asiento del conductor Elementos: <ul style="list-style-type: none">• Conductor• Cámara Digital
	Solución del Requerimiento: La cámara se posicionará para capturar el rostro del conductor en video como entrada para el sistema.

Tabla 11: RF02- Capturar Video

ID	Nombre corto del Requerimiento
RF02	<i>Capturar video</i>
	Descripción: El sistema capturará video en todo momento. Elementos: <ul style="list-style-type: none">• Conductor• Cámara Digital
	Solución del Requerimiento: Con el uso de una cámara digital, se capturará video en tiempo real.

Tabla 12: RF03- Detectar Características Faciales

ID	Nombre corto del Requerimiento
RF03	<i>Detectar Características Faciales</i>
Descripción: El sistema detectará las características faciales más relevantes; rostro, ojos y boca del conductor, mediante el uso de un método de detección facial.	
Elementos: <ul style="list-style-type: none"> • Imagen preprocesada • Método de Detección Facial 	
Solución del Requerimiento: El sistema obtendrá la posición del rostro y, en base a esta, las posiciones de la boca y los ojos del conductor detectados.	

Tabla 13: RF04- Clasificar imagen

ID	Nombre corto del Requerimiento
RF04	<i>Clasificar imagen</i>
Descripción: El sistema evaluará los signos de somnolencia, como el estado de los ojos y la apertura de la boca para clasificar la imagen en base a la presencia o ausencia de somnolencia mediante técnicas de aprendizaje automático y/o aprendizaje profundo. Así mismo se escalarán y se normalizarán las imágenes que servirán como entrada para dicho proceso.	
Elementos: <ul style="list-style-type: none"> • Técnicas de aprendizaje automático y/o aprendizaje profundo. 	
Solución del Requerimiento: El sistema obtendrá el estado de somnolencia del conductor.	

Tabla 14: RF05- Enviar señal de Alerta

ID	Nombre corto del Requerimiento
RF05	<i>Enviar señal de alerta</i>
Descripción: Se enviará una señal de alerta al módulo central de procesamiento cuando se detecten signos de somnolencia en el conductor.	
Elementos: <ul style="list-style-type: none"> • Microordenador • Zumbador 	
Solución del Requerimiento: El módulo Central de procesamiento activará la alarma al recibir la señal de alerta.	

Tabla 15: RF06- Almacenar Incidencia

ID	Almacenar Incidencia
<i>RF06</i>	<i>Enviar señal de alerta</i>
Descripción:	Se almacenará un fragmento del video en caso de detectar somnolencia en el conductor.

Elementos:

- Microordenador
- Memoria Externa
- Video

Solución del Requerimiento:

Se guardará el video de incidencia en caso de no contar con conexión inalámbrica, para ser enviada posteriormente a un servicio almacenamiento en la nube.

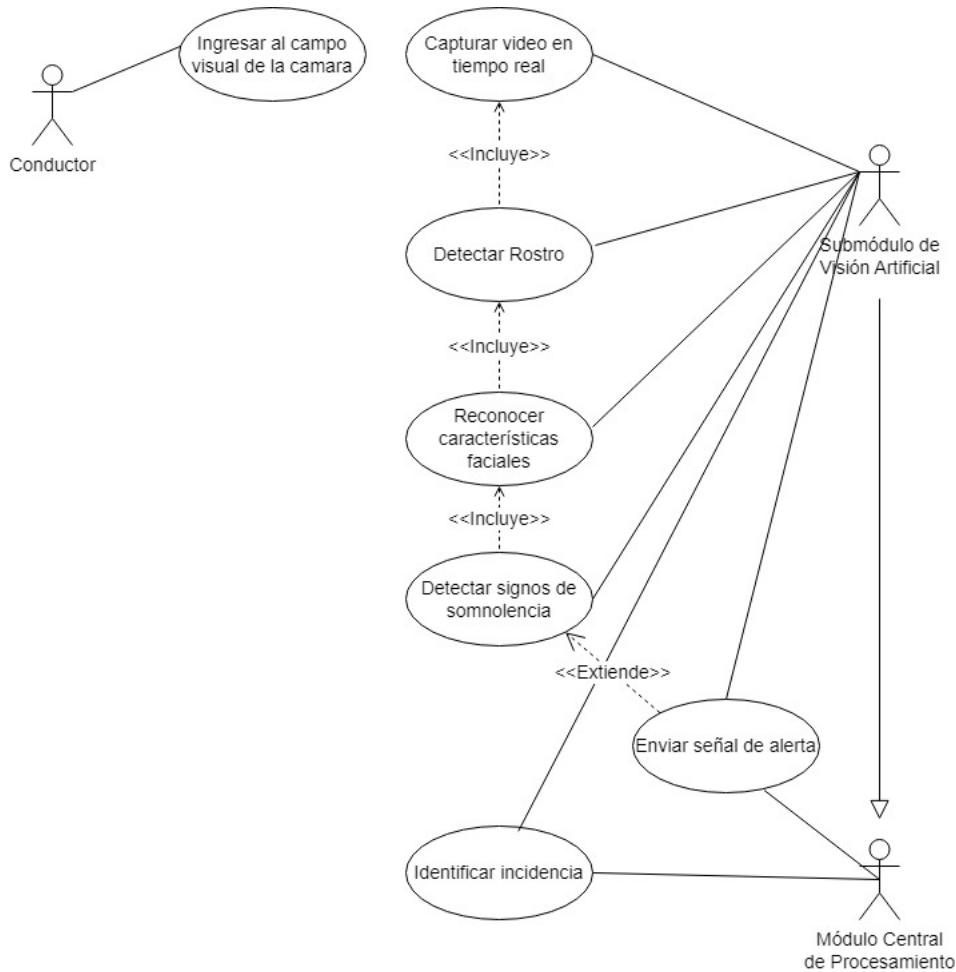


Figura 9: Diagrama de Casos de Uso - Submódulo de Visión Artificial

Especificación de Casos de Uso

Tabla 16: Caso de Uso 01 - Ingresar al campo visual de la Cámara

Nombre del Caso de Uso	<i>Ingresar al campo visual de la Cámara</i>
Actores	<i>Conductor</i>
Descripción: El conductor ingresará al campo visual de la cámara al entrar al auto	
Precondiciones:	
<ul style="list-style-type: none"> • El conductor deberá encender el auto 	
Entradas:	
<ul style="list-style-type: none"> • Conductor • Automóvil 	

Tabla 17: Caso de Uso 02 - Capturar Video en Tiempo real

Nombre del Caso de Uso	<i>Capturar Video en Tiempo real</i>
Actores	<i>Submódulo de Visión Artificial</i>
Descripción: El submódulo de visión artificial capturara un video en tiempo real utilizando una cámara digital	
Precondiciones:	
<ul style="list-style-type: none"> • El conductor deberá encender el auto 	
Entradas:	
<ul style="list-style-type: none"> • Conductor • Automóvil • Cámara Digital 	

Tabla 18: Caso de Uso 03 - Detectar Rostro

Nombre del Caso de Uso	<i>Detectar Rostro</i>
Actores	<i>Submódulo de Visión Artificial</i>
Descripción: El submódulo de visión artificial detectará el rostro del conductor.	
Precondiciones:	
<ul style="list-style-type: none"> • El conductor deberá permanecer en el campo visual de la cámara 	
Entradas:	
<ul style="list-style-type: none"> • Conductor • Automóvil • Cámara Digital • Rostro 	

Tabla 19: Caso de Uso 04 - Reconocer características faciales

Nombre del Caso de Uso	<i>Reconocer características faciales</i>
Actores	<i>Submódulo de Visión Artificial</i>
Descripción: El submódulo de visión artificial reconocerá características faciales tales como: ojos, boca	<p>Precondiciones:</p> <ul style="list-style-type: none"> • El sistema deberá haber detectado previamente el rostro del conductor <p>Entradas:</p> <ul style="list-style-type: none"> • Conductor • Ojos • Cámara Digital • Boca • Rostro

Tabla 20: Caso de Uso 05 - Detectar Signos de Somnolencia

Nombre del Caso de Uso	<i>Detectar Signos de Somnolencia</i>
Actores	<i>Submódulo de Visión Artificial</i>
Descripción: El submódulo de visión artificial detectará signos de somnolencia utilizando métricas de tiempo para los ojos y tomará en cuenta la apertura de la boca	<p>Precondiciones:</p> <ul style="list-style-type: none"> • El sistema deberá haber extraído previamente la información de los ojos y boca <p>Entradas:</p> <ul style="list-style-type: none"> • Conductor • Ojos • Cámara Digital • Boca • Rostro

Tabla 21: Caso de Uso 06 - Enviar Señal de Alerta

Nombre del Caso de Uso	<i>Enviar Señal de Alerta</i>
Actores	<i>Submódulo de Visión Artificial, Módulo Central de Procesamiento</i>
Descripción: El submódulo de visión artificial enviará una alerta al Módulo Central de Procesamiento para activar la alarma	
Precondiciones:	
<ul style="list-style-type: none"> • El sistema deberá haber detectado signos de somnolencia en el conductor 	
Entradas:	
<ul style="list-style-type: none"> • Conductor • Ojos • Cámara Digital • Boca • Alerta 	

Tabla 22: Caso de Uso 07 - Identificar Incidencia

Nombre del Caso de Uso	<i>Identificar Incidencia</i>
Actores	<i>Submódulo de Visión Artificial</i>
Descripción: El submódulo realizará un reporte de incidencia que contenga la fecha hora y lugar de la incidencia	
Precondiciones:	
<ul style="list-style-type: none"> • El sistema deberá haber extraído previamente la información de los ojos y boca 	
Entradas:	
<ul style="list-style-type: none"> • Conductor • Ojos • Cámara Digital • Boca • Rostro 	

6.2.6. Análisis y Elección de herramientas de programación

- **OpenCV**

Es una biblioteca de procesamiento de imágenes de código abierto, ampliamente utilizada en la investigación académica y la industria. Permite el análisis de imágenes, la detección de objetos, el reconocimiento de patrones y la segmentación de imágenes.

- **Numpy**

Es una librería de Python especializada en el cálculo numérico y el análisis de datos, especialmente para un gran volumen de datos. La ventaja de Numpy frente a las listas predefinidas en Python es que el procesamiento de los arrays se realiza mucho más rápido (hasta 50 veces más) que las listas, lo cual la hace ideal para el procesamiento de vectores y matrices de grandes dimensiones [41].

6.2.7. Análisis y Elección de la cámara digital

Para la búsqueda de modelos de cámaras digitales, se buscaron modelos que tuvieran compatibilidad con la Nvidia Jetson Nano debido a que el sistema será alojado en este micrordenador. Se optó por utilizar la cámara IMX219-160, ya que cuenta con hasta 8 Megapixeles de resolución, además de tener distintos los formatos de salida jpg, RAW, DNG O MP4. También cuenta con diferentes resoluciones de video que van desde 1080p hasta 480p a 30 cuadros por segundo.



Figura 10: Modulo de Cámara IMX219-160

6.2.8. Análisis y Elección de algoritmos de visión artificial

Existen varios métodos y bibliotecas utilizadas para detectar rostros en imágenes y videos en tiempo real, algunos de los más comunes son:

- **OpenCV con Haar Cascade:** Utilizar el clasificador Haar Cascade de OpenCV para la detección facial en tiempo real. Este método es rápido y puede implementarse fácilmente.
- **Face Recognition Library:** Utiliza bibliotecas específicas para reconocimiento facial, como la biblioteca face_recognition en Python, que utiliza el detector HOG de dlib.
- **MTCNN (Multi-Task Cascaded Convolutional Networks):** Implementa MTCNN para la detección facial y la localización de puntos de referencia. Este método es eficiente y puede detectar múltiples rostros en una sola imagen.
- **SSD (Single Shot Multibox Detector):** Emplea modelos SSD para la detección de objetos en tiempo real, incluidos los rostros. Utiliza bibliotecas como TensorFlow o PyTorch con modelos SSD preentrenados.
- **YOLO (You Only Look Once):** Utiliza YOLO para la detección de objetos, incluidos los rostros. YOLO es conocido por su velocidad y eficiencia en tiempo real.

Debido a los requerimientos del sistema, Face Recognition Library, utiliza el detector HOG de dlib, es una elección sólida para detectar el rostro, los ojos y la boca. Sus principales ventajas incluyen una detección precisa de rostros, la capacidad de localizar puntos faciales específicos, flexibilidad y facilidad de uso.

6.3. Análisis del Módulo de Comunicaciones

Los sistemas automatizados de comunicación, mayormente inalámbricos, se encargan de recopilar datos remotos y transmitir la información. La tecnología elegida para la implementación de este módulo es la red 4G/LTE, su estándar es establecido por el 3GPP (Proyecto de asociación de tercera generación) donde establece sus especificaciones en la versión 8 de sus estándares, el cual clasifica hasta 13 categorías LTE [42]. En estas categorías se especifica la velocidad máxima de carga y descarga, siendo la categoría 0 la velocidad más baja. En la Tabla 23 se observan las categorías LTE que existen junto con su velocidad máxima de bajada y subida, de la categoría 0 a la 5 se definen para equipos de usuarios, es decir, telefonía celular.

Tabla 23: Categorías LTE

Categoría	Velocidad Máxima de Bajada(Mbps)	Velocidad Máxima de Subida (Mpbs)
0	1	1
1	10	5
2	50	25
3	100	50
4	150	50
5	300	75
6	300	50
7	300	150
8	1200	600
9	450	50
10	450	100
11	600	50
12	600	100
13	390	150

Para este sistema, el análisis debe cumplir con los parámetros que dicta el teorema de Shannon-Hartley establece que, dado un canal con ruido con una capacidad C e información transmitida en una tasa R entonces si $R < C$ existe una técnica de codificación que permite que la probabilidad de error en el Rx se reduzca. Se debe cumplir que la tasa de bits debe ser siempre menor a la capacidad del canal.

Este análisis se divide en:

- *Telemetría:*

El módulo de telemetría es el encargado de recopilar, procesar y transmitir las coordenadas de la ubicación geográfica del conductor a la estación base que se encarga de monitorear los datos obtenidos.

- *Datos:*

El encargado de recopilar, procesar y transmitir los fotogramas que el sistema identifica como somnolencia en el conductor, así como un mensaje informativo.

- *Cobertura:*

Encargado de revisar la cobertura de la zona o regiones presentes en el alcance del sistema.

6.3.1. Análisis del Submódulo de Telemetría

El objetivo que tiene el módulo de telemetría es mandar el posicionamiento mediante coordenadas (latitud y longitud) del vehículo en movimiento en tiempo real. Las coordenadas serán mandadas en forma de cadena de texto, teniendo un aproximado de hasta 300 bits, otro punto a considerar es el periodo de tiempo entre el envío de cada posición, al tratarse de un vehículo en movimiento y la velocidad máxima siendo regida por Semovi (Secretaría de movilidad de la ciudad de México) que dicta una velocidad promedio de entre 50-80 km/h [43].

Se propone una velocidad de 60 km/h:

$$60 \frac{\text{km}}{\text{h}} \cdot \frac{1000\text{m}}{1\text{km}} \cdot \frac{1\text{h}}{60\text{min}} \cdot \frac{1\text{min}}{60\text{s}} = 16,6 \frac{\text{m}}{\text{s}} \approx 17 \frac{\text{m}}{\text{s}}$$

El automóvil se desplaza aproximadamente 17 metros en 1 segundo por lo que al realizar el envío en periodos de 10 segundos, se garantiza el trazado de la ruta en el mapa. Teniendo 170 metros recorridos cada 10 segundos del viaje. Verificando que se cumpla el Teorema de Shannon-Hartley tenemos:

$$300\text{bits} \cdot 10\text{s} = 3\text{kbps}$$

y la máxima tasa de datos que el transceptor tiene es de 5 Mbps.

Cumpliéndose la relación establecida de $R < C$.

$$3\text{kbps} < 5\text{Mbps}$$

6.3.2. Análisis del Submódulo de Datos

El objetivo que tiene el análisis de los datos es enviar fotogramas a la estación base para su futura gestión y validación, así como el almacenamiento de estos; de igual manera, se pretende el envío de mensajes informativos a la estación base.

Para la transmisión del material multimedia a utilizar, se propone que el clip grabado tenga una profundidad de bits de mínimo 8 y una resolución de mínimo 960 x 540 píxeles, comúnmente conocido como QHD o "Quarter of High Definition" por sus siglas en inglés, todo esto en una secuencia de 30 fps.

Tenemos que:

$$960 \times 540 \times 8 = 4,147,200 \text{ bits}$$

Cada imagen tiene 4,147,200 bits entonces en 1 segundo se transmiten:

$$4,147,200 \times 30\text{fps} = 124,41 \text{ Mbps}$$

Que rebasa la capacidad del canal de red establecida, por lo que se implementa el uso de un compresor de video MP4 para así modificar la resolución en la que se almacena el video en la estación base. Por otro lado, se envían al menos 2 fotogramas por segundo para no sobrepasar la capacidad del canal para este caso.

$$640 \times 480 \times 8 = 2,457,600 \text{ bits}$$

Cada imagen tiene 2,457,600 bits entonces en 1 segundo se transmiten:

$$2,457,600 \times 2\text{fps} = 4,91 \text{ Mbps}$$

Cumpliéndose nuevamente la relación establecida $R < C$.

$$4,91 Mbps < 5 Mbps$$

Más 3 kbps de la tasa de bits a utilizar del texto informativo.

6.3.3. Análisis del Submódulo de Cobertura

La red de telefonía celular funciona mediante celdas, figura 11), estas celdas tienen el objetivo de mantener conectados a los dispositivos en las diferentes áreas de cobertura y cada una de ellas contiene una estación base o antena. El *handover* o *handoff* es el proceso de transferir el servicio de una celda a otra[44] y se tiene la siguiente métrica:

- Se considera un *handoff* fuerte cuando al ir cambiando de celdas se presente algún problema en la velocidad de la red logrando así la pérdida de la misma.
- Se considera un *handoff* suave cuando al ir cambiando de celdas no se presente algún problema de la red o este sea mínimo.

Ya sea cuando se pierde la calidad en una de ellas o cuando el dispositivo se va trasladando. Este mecanismo tiene por objetivo garantizar la correcta realización del servicio en las condiciones mencionadas anteriormente. Para el análisis de cobertura de la red LTE se tiene en cuenta el sistema *handoff* porque el dispositivo se va trasladando entre distintas celdas.

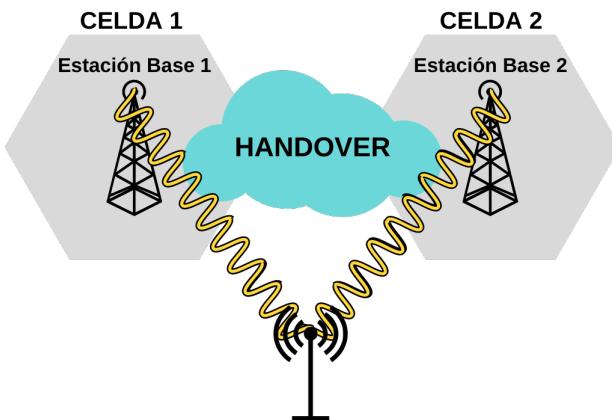


Figura 11: Esquema *handover* entre dos celdas

La cobertura de LTE por Telcel es la más amplia del país [4], su mayor cobertura está concentrada en la Ciudad de México, por lo que existe una buena recepción, aunque en ocasiones debido a la saturación de la red o al transferir el servicio de la antena de una estación base a otra se puede encontrar alguna intermitencia o retraso, en estos casos el módulo va a pasar a modo de espera para enviar la información una vez se restablezca la red.



Figura 12: Mapa de cobertura de la red LTE

En el mapa de cobertura la zona verde hace referencia a zonas con cobertura y la zona amarilla cobertura no garantizada, en caso de realizar pruebas en las zonas amarillas del mapa se va a notar un cambio en la velocidad del dispositivo móvil ya sea para la transferencia de archivos o la conectividad a la red ocasionando un *handoff fuerte*; por el otro lado, al estar realizando pruebas en la zona verde del mapa su *handoff* será más suave.

6.4. Análisis del Módulo de Estación Base

El módulo de la estación base tiene como objetivo que el usuario administrador, visualice y confirme el estado de los reportes de incidencia que se hayan presentado por parte del conductor, por tal motivo, se realizará una aplicación web, la cual se conectará a una base de datos NoSQL. En ella se guardarán los reportes de incidencias y se podrán visualizar por medio de la aplicación web. Cabe aclarar que el video de la incidencia se almacenará en la nube, ya que al ser contenido multimedia no se puede guardar en la base de datos, únicamente se guardará el URL para acceder al video. Las credenciales del usuario serán almacenadas en Amazon Cognito, con la cuales podrán iniciar sesión dentro de la Aplicación Web, la cual estará alojada en un servidor web.

El sistema contará con los siguientes requerimientos:

Requerimientos Funcionales del Módulo de Estación Base

Tabla 24: RF01- Guardar Incidencia

ID	Nombre corto del Requerimiento
RF01	<p>Guardar Incidencia</p> <p>Descripción: La base de datos almacenará la información de cada reporte de incidencia que se envíe desde el Módulo Central de Procesamiento.</p> <p>Elementos:</p> <ul style="list-style-type: none">• Id del Conductor• Nombre de Conductor• Apellidos de Conductor• Número de Incidencias• Fecha• Hora• Estado de la Incidencia• Ubicación• URL del video <p>Solución del Requerimiento: El Módulo de Procesamiento Central puede insertar datos en la base de datos.</p>

Tabla 25: RF02- Guardar Video

ID	Nombre corto del Requerimiento
RF02	<i>Guardar Video</i>
	<p>Descripción: La base de datos almacenará la información de cada reporte de incidencia que se envíe desde el Módulo Central de Procesamiento.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Video de incidencia <p>Solución del Requerimiento:</p> <p>El Módulo de Procesamiento Central puede insertar el video en el almacenamiento de objetos, mientras que la base de datos obtiene la URL del video guardado.</p>

Tabla 26: RF03- Conectar Aplicación Web

ID	Nombre corto del Requerimiento
RF03	<i>Conectar Aplicación Web</i>
	<p>Descripción: La base de datos almacenará la información de cada reporte de incidencia que se envíe desde el Módulo Central de Procesamiento.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Node.js • React • Express • MongoDB <p>Solución del Requerimiento:</p> <p>El usuario administrador puede realizar la inserción, modificación, eliminación y consulta de datos desde la aplicación web.</p>

Tabla 27: RF04- Desplegar Aplicación Web

ID	Nombre corto del Requerimiento
RF04	<i>Desplegar Aplicación Web</i>
	<p>Descripción: La aplicación web se alojará en una red de entrega de contenido (CDN), disponible con una URL.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Content Delivery Network <p>Solución del Requerimiento:</p> <p>El usuario administrador puede acceder a la interfaz de la aplicación web, haciendo uso de la URL en el navegador web.</p>

Tabla 28: RF05- Guardar Credenciales de Usuario Administrador

ID	Nombre corto del Requerimiento
RF05	Guardar Credenciales de Usuario Administrador
Descripción: Se guardarán únicamente las credenciales de los usuarios administradores que podrán acceder a la aplicación web.	
Elementos:	
<ul style="list-style-type: none"> • Servicio de Administración de Acceso e Identidad 	
Solución del Requerimiento:	
El usuario administrador podrá iniciar sesión en la aplicación web utilizando un servicio de Administración de Acceso e Identidad	

Requerimientos No Funcionales del Módulo de Estación Base

Tabla 29: Requerimientos No Funcionales del Módulo de Estación Base

ID	Nombre del requerimiento	Descripción
RNF01	Disponibilidad	La disponibilidad del sistema será continua, el usuario podrá acceder a la información las 24 horas del día.
RNF02	Interoperabilidad	El sistema será capaz de intercambiar información con el Módulo Central de Procesamiento a través del Módulo de Telemetría.
RNF03	Seguridad	El sistema hará uso de encriptación AES-256 para garantizar la seguridad de las credenciales de los usuarios.
RNF04	Usabilidad	El sistema estará enfocado a la visualización de reportes, por lo que este tendrá una interfaz intuitiva y amigable para el usuario. El sistema proporcionará mensajes de advertencia orientados al usuario, en caso de ocurrir un error en el Módulo Central de Procesamiento.

Análisis de Casos de Uso

Con base en los requerimientos se realizó el siguiente diagrama de casos de uso, el cual muestra las actividades y la interacción con el Módulo de Estación base y el Módulo Central de Procesamiento.

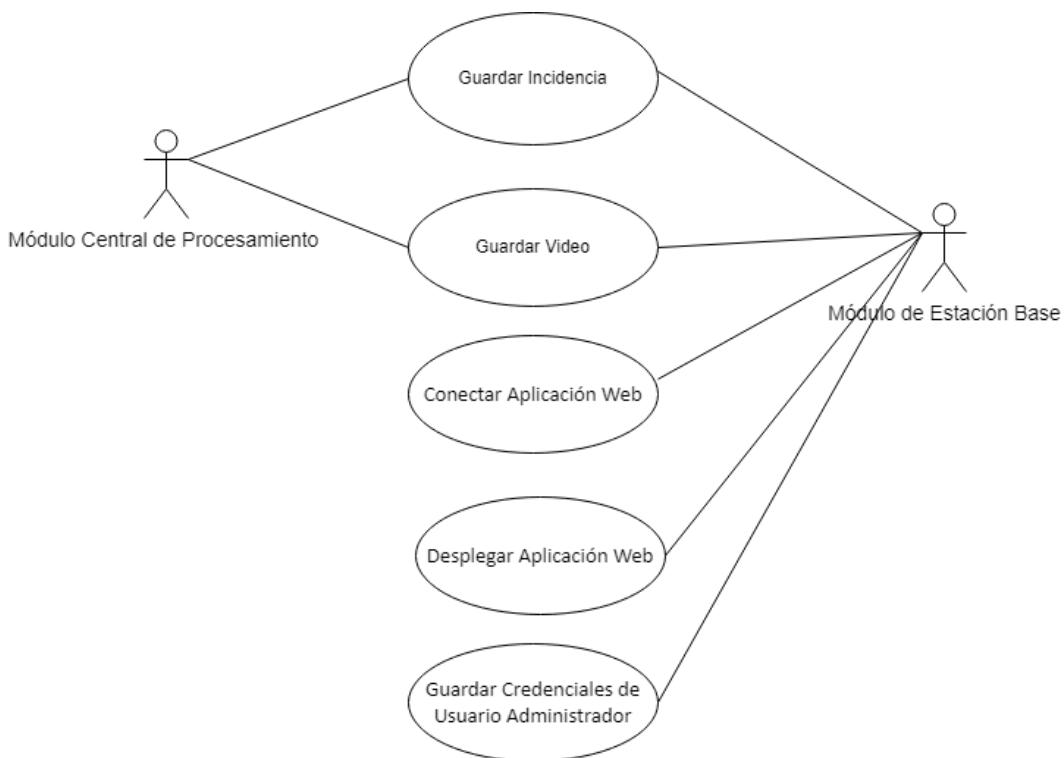


Figura 13: Diagrama de Casos de Usos del Módulo de Estación Base

Tabla 30: Caso de Uso 01 - Guardar Incidencia

Nombre del Caso de Uso	<i>Guardar Incidencia</i>
Actores	<i>Módulo Central de Procesamiento, Módulo de Estación Base</i>
Descripción: El sistema almacenará las incidencias recibidas por el módulo de Comunicaciones.	
Precondiciones: <ul style="list-style-type: none"> • Contar con una conexión al Módulo de Comunicaciones • Contar con una conexión a la Base de Datos Elementos: <ul style="list-style-type: none"> • Reporte de Incidencia • Módulo de Comunicaciones 	

Tabla 31: Caso de Uso 02 - Guardar Video

Nombre del Caso de Uso	<i>Guardar Video</i>
Actores	<i>Módulo Central de Procesamiento Módulo de Estación Base</i>
Descripción: Se almacenará el video recibido por el Módulo de Comunicaciones	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con una conexión al Módulo de Comunicaciones • Contar con una conexión a la Base de Datos 	
Elementos:	
<ul style="list-style-type: none"> • Video • Módulo de Comunicaciones 	

Tabla 32: Caso de Uso 03 - Conectar Aplicación Web

Nombre del Caso de Uso	<i>Conectar Aplicación Web</i>
Actores	<i>Módulo de Estación Base</i>
Descripción: Se realizará la conexión de la aplicación web y la base de datos	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con el servicio de Hosting Disponible 	
Elementos:	
<ul style="list-style-type: none"> • Aplicación Web • Base de Datos • Servicio de Hosting 	

Tabla 33: Caso de Uso 04 - Desplegar Aplicación Web

Nombre del Caso de Uso	<i>Desplegar Aplicación Web</i>
Actores	<i>Módulo Central de Procesamiento, Módulo de Estación Base</i>
Descripción: Utilizando el servicio de Hosting ofrecido por Amazon Amplify se desplegará la aplicación web	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con una cuenta registrada en Amazon Amplify 	
Elementos:	
<ul style="list-style-type: none"> • Aplicación Web • Base de Datos • Servicio de Hosting 	

Tabla 34: Caso de Uso 05 - Guardar Credenciales de Usuario Administrador

Nombre del Caso de Uso	<i>Guardar Credenciales de Usuario Administrador</i>
Actores	<i>Módulo Central de Procesamiento</i>
Descripción:	Utilizando Amazon Congito, el sistema permitirá almacenar nuevos usuarios administradores
Precondiciones:	<ul style="list-style-type: none"> • Contar con el servicio de la aplicación web Desplegado
Elementos:	<ul style="list-style-type: none"> • Amazon Cognito

6.4.1. Análisis de la Aplicación Web

A continuación se listan los requerimientos Funcionales obtenidos

- Análisis de Requerimientos Funcionales

Tabla 35: RF01- Iniciar Sesión

ID	Nombre corto del Requerimiento
RF01	<i>Iniciar Sesión</i>
Descripción:	El sistema permitirá iniciar sesión en la aplicación web
Elementos:	<ul style="list-style-type: none"> • Servicio de Administración de Acceso e Identidad • Administrador • Credenciales • Gestor de Base de Datos
Solución del Requerimiento:	<p>El sistema, con la ayuda un servicio de administración de acceso e identidad del cliente, comprobará que las credenciales ingresadas por parte del usuario administrador se encuentren en la base de datos. En caso contrario, la aplicación web indicará que ese usuario no se encuentra registrado en la base de datos.</p>

Tabla 36: RF02- Mostrar el Historial de Reportes de Incidencia

ID	Nombre corto del Requerimiento
RF02	<i>Mostrar el Historial de Reportes de Incidencia</i>
Descripción: El sistema desplegará en forma de lista todas las incidencias que se tengan registradas en la base de datos	
Elementos:	
<ul style="list-style-type: none"> • Conductor • Incidencias • Administrador • Gestor de Base de Datos 	
Solución del Requerimiento:	
El sistema recuperará de la base de datos todas las incidencias que se tengan registradas.	

Tabla 37: RF03- Visualizar Reporte de Incidencia

ID	Nombre corto del Requerimiento
RF03	<i>Visualizar Reporte de Incidencia</i>
Descripción: El sistema desplegará los detalles específicos de cada incidencia registrada.	
Elementos:	
<ul style="list-style-type: none"> • Incidencia • Administrador • Gestor de Base de Datos 	
Solución del Requerimiento:	
El usuario administrador podrá visualizar los reportes individuales de incidencias de cada conductor al hacer click en cualquiera de las incidencias mostrada en la lista principal. Los reportes contendrán información sobre la fecha, hora, ubicación y un video corto del momento en que fueron detectados síntomas de somnolencia.	

Tabla 38: RF04- Confirmar Incidencia

ID	Nombre corto del Requerimiento
RF04	<i>Confirmar Incidencia</i>
<p>Descripción: El sistema permitirá al administrador confirmar la incidencia, esto para descartar que se trate de un falso positivo.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Administrador • Incidencia • Conductor • Gestor de Base de Datos 	
<p>Solución del Requerimiento:</p> <p>El usuario administrador podrá confirmar la incidencia después de haber revisado el videoclip del momento de somnolencia con la intención de descartar falsos positivos. Esto será posible ingresando a una incidencia específica mostrando sus detalles.</p>	

Tabla 39: RF05 - Recuperar Contraseña

ID	Nombre corto del Requerimiento
RF05	<i>Recuperar Contraseña</i>
<p>Descripción: El sistema contará con una opción para recuperar la contraseña del administrador en caso de que sea olvidada la contraseña.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Servicio de Administración de Acceso e Identidad • Email • Administrador • Gestor de Base de Datos 	
<p>Solución del Requerimiento:</p> <p>El sistema requerirá que el usuario administrador ingrese el correo con el que fue registrado. Posteriormente se le enviará un código de recuperación de contraseña a ese correo. El administrador ingresará ese código en el apartado de recuperar contraseña y así podrá ingresar una nueva contraseña.</p>	

Tabla 40: RF06- Mostrar perfil del Conductor

ID	Nombre corto del Requerimiento
RF06	<i>Mostrar Perfil Conductor</i>
Descripción: El sistema permitirá al usuario administrador visualizar los datos de cada conductor registrado en la base de datos.	
Elementos: <ul style="list-style-type: none"> • Perfil • Conductor • Gestor de Base de Datos 	
Solución del Requerimiento: El administrador podrá consultar cada uno de los perfiles de los conductores registrados en la base de datos dando click en el nombre del mismo. En dicho perfil se mostrarán datos como nombre, apellido, así como el número de incidencias de dicho conductor, con sus respectivos detalles.	

Tabla 41: RF07- Mostrar ubicación Geográfica

ID	Nombre corto del Requerimiento
RF07	<i>Mostrar Ubicación Geográfica</i>
Descripción: La aplicación web, con ayuda de los datos proporcionados por el módulo de telemetría, mostrará la ubicación en tiempo real de un conductor.	
Elementos: <ul style="list-style-type: none"> • Ubicación en Tiempo Real • Módulo de Telemetría 	
Solución del Requerimiento: El sistema permitirá al administrador consultar la ubicación de los conductores en tiempo real. Para esto el administrador deberá de ingresar previamente al perfil del conductor del cual desea consultar dicha ubicación.	

Tabla 42: RF08- Descartar Incidencia

ID	Nombre corto del Requerimiento
RF08	<i>Descartar Incidencia</i>
Descripción: El sistema permitirá catalogar una incidencia como Falsa si fuera el caso.	
Elementos: <ul style="list-style-type: none"> • Incidencia • Administrador 	
Solución del Requerimiento:	
En caso de presentarse una incidencia falsa, después de haber sido revisada por el administrador, esta podrá ser catalogada como falsa incidencia y será eliminada automáticamente de la base de datos.	

Tabla 43: RF09- Registrar Usuario

ID	Nombre corto del Requerimiento
RF09	<i>Registrar Usuario</i>
Descripción: El sistema le permitirá al administrador registrar nuevos conductores.	
Elementos: <ul style="list-style-type: none"> • Nombre • Apellido • ID 	
Solución del Requerimiento:	
El Administrador podrá registrar a nuevos conductores en el sistema. El sistema generará de manera automática el ID del nuevo conductor	

Tabla 44: RF10- Mostrar el Historial de Reportes de Incidencia

ID	Nombre corto del Requerimiento
RF10	<i>Modificar Usuario</i>
Descripción: El sistema le permitirá al administrador modificar los datos del conductor.	
Elementos: <ul style="list-style-type: none"> • Nombre • Apellido • Conductor • Administrador 	
Solución del Requerimiento:	
El administrador podrá editar los datos de los conductores como nombre o apellido.	

Tabla 45: RF11- Mostrar el Historial de Reportes de Incidencia

ID	Nombre corto del Requerimiento
RF11	<i>Eliminar Usuario</i>
	<p>Descripción: El sistema le permitirá al administrador eliminar los datos del conductor.</p> <p>Elementos:</p> <ul style="list-style-type: none"> • Conductor • Administrador
	<p>Solución del Requerimiento:</p> <p>El sistema le permitirá al eliminar los datos de conductores de la base de datos.</p>

■ Análisis de Requerimientos no Funcionales

A continuación se mencionan los requerimientos no funcionales de la aplicación web.

Tabla 46: Requerimientos No funcionales - Aplicación Web

Requerimiento No Funcional	Descripción
Disponibilidad	La aplicación web deberá estar disponible las 24 horas del día
Interoperabilidad	La aplicación deberá ser accesible desde cualquier sistema operativo mientras se cuenta con un navegador web en su versión más actual
Eficiencia	La aplicación tendrá tiempos de respuesta menores a 100ms.
Escalabilidad	La aplicación web deberá ofrecer la oportunidad de agregar nuevas funcionalidades y soportar un mayor número de usuarios a futuro
Estabilidad	La aplicación deberá ofrecer un buen funcionamiento mientras se tenga una conexión a internet mínima de 1Mbps

cupera

■ Análisis de Casos de Uso

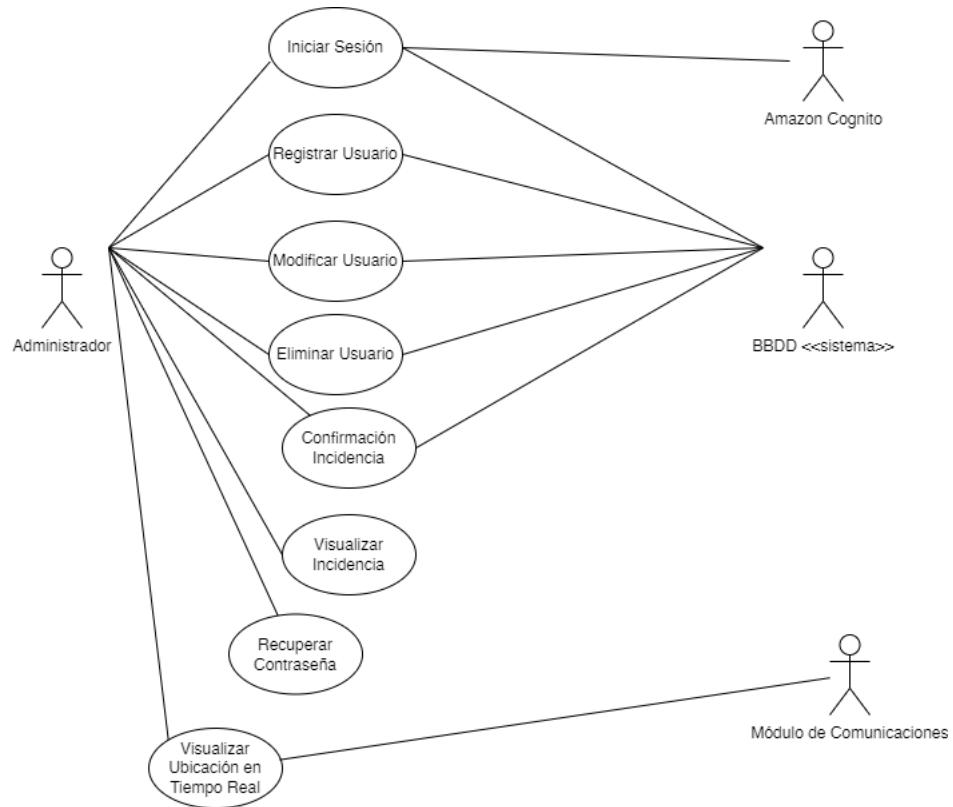


Figura 14: Diagrama de Casos de Uso - Aplicación Web

Especificación de Casos de Uso

Tabla 47: Caso de Uso 01 - Iniciar Sesión

Nombre del Caso de Uso	<i>Iniciar Sesión</i>
Actores	<i>Administrador, BBDD, Amazon Cognito</i>
Descripción: El administrador ingresará los datos correspondientes para iniciar sesión en la aplicación web.	
Precondiciones:	
<ul style="list-style-type: none"> • El administrador deberá contar sus credenciales registradas en Amazon Cognito 	
Entradas:	
<ul style="list-style-type: none"> • Email • Contraseña 	

Tabla 48: Caso de Uso 02 - Registrar Usuario

Nombre del Caso de Uso	<i>Registrar Usuario</i>
Actores	<i>Administrador, BBDD</i>
Descripción: Solo los usuarios administradores podrán registrar a nuevos conductores en el sistema.	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de acceso como administrador 	
Entradas:	
<ul style="list-style-type: none"> • Nombre • Apellido • Edad 	

Tabla 49: Caso de Uso 03 - Modificar Usuario

Nombre del Caso de Uso	<i>Modificar Usuario</i>
Actores	<i>Administrador, BBDD</i>
Descripción: El usuario Administrador podrá modificar los datos de los conductores previamente registrados.	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de Administrador • Que exista una entrada del conductor a modificar 	
Entradas:	
<ul style="list-style-type: none"> • Nombre • Apellido • Edad 	

Tabla 50: Caso de Uso 04 - Eliminar Usuario

Nombre del Caso de Uso	<i>Eliminar Usuario</i>
Actores	<i>Administrador, BBDD</i>
Descripción: El usuario administrador podrá dar de baja a un conductor	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de Administrador • Que exista una entrada del conductor a eliminar 	
Entradas:	
<ul style="list-style-type: none"> • ID del conductor 	

Tabla 51: Caso de Uso 05 - Confirmar Incidencia

Nombre del Caso de Uso	<i>Confirmar Incidencia</i>
Actores	<i>Administrador, BBDD</i>
Descripción: El administrador podrá confirmar las incidencias	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de Administrador • Haber iniciado sesión previamente 	
Entradas:	
<ul style="list-style-type: none"> • Correo • Contraseña 	

Tabla 52: Caso de Uso 06 - Visualizar Incidencia

Nombre del Caso de Uso	<i>Visualizar Incidencia</i>
Actores	<i>Administrador, BBDD</i>
Descripción: El administrador podrá reproducir un video dónde se muestre el momento en que ocurrió la incidencia	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de administrador • Haber seleccionado una incidencia 	
Entradas	
<ul style="list-style-type: none"> • ID incidencia 	

Tabla 53: Caso de Uso 07 - Recuperar Contraseña

Nombre del Caso de Uso	<i>Recuperar Contraseña</i>
Actores	<i>Administrador, BBDD</i>
Descripción: El administrador podrá recuperar contraseña en caso de olvidarla	
Precondiciones:	
<ul style="list-style-type: none"> • Contar con credenciales de administrador 	
Entradas	
<ul style="list-style-type: none"> • Email • Nueva Contraseña 	

Tabla 54: Caso de Uso 08 - Visualizar Ubicación en Tiempo Real

Nombre del Caso de Uso	<i>Visualizar Ubicación en Tiempo Real</i>
Actores	<i>Administrador, Módulo de Comunicaciones</i>
Descripción:	El administrador podrá visualizar la ubicación en tiempo real de los conductores registrados
Precondiciones:	<ul style="list-style-type: none"> • Contar con credenciales de Administrador • Haber iniciado sesión previamente
Entradas:	<ul style="list-style-type: none"> • ID del conductor

6.4.2. Análisis y Elección del manejador de bases de datos

Debido a la escalabilidad horizontal, facilidad de implementación, bajo coste y flexibilidad que ofrecen los gestores de bases de datos NoSQL para adaptarse a las necesidades del proyecto, se optó por la elección de un gestor de este tipo. Además, se contemplaron las posibles modificaciones dentro del modelado de la base de datos, las cuales tendrán como motivo cumplir los requisitos del sistema y presentar la información necesaria en la aplicación web. Las modificaciones necesarias podrán ser fácilmente aplicadas usando un SGBD NoSQL, ya que ofrecen una mayor facilidad de realizar cambios dentro del esquema, a diferencia de un SGBD SQL.

Se eligió Amazon DynamoDB como SGBD NoSQL debido a su facilidad de integración con los servicios de autenticación y despliegue de AWS Amplify que se usarán para el alojamiento de la aplicación 6.4.4. DynamoDB ofrece una solución escalable y totalmente administrada, lo que complementa eficientemente la arquitectura de la aplicación, permitiendo una gestión sencilla y eficaz de los datos junto con características clave de autenticación y despliegue proporcionadas por AWS Amplify.

6.4.3. Análisis y Elección de lenguajes de programación web

Al igual que en el desarrollo del *backend*, existen diversas opciones para el desarrollo del frontend, e incluso en mayor medida. En este caso, los elementos fundamentales para el desarrollo del frontend de una aplicación son HTML, JavaScript y CSS.

JavaScript ha experimentado una evolución significativa y cuenta con una comunidad considerable, lo cual lo posiciona como uno de los principales lenguajes para el desarrollo frontend.

A continuación se describen sus características más importantes:

- **Versatilidad**

JavaScript es un lenguaje que permite crear interactividad y dinamismo en las aplicaciones web. Su capacidad para manipular el contenido y la apariencia de una página en tiempo real, así como interactuar con elementos del DOM (Document Object Model), lo convierte en una elección ideal para crear interfaces de usuario interactivas y atractivas.

- **Multiplataforma** Javascript puede ser ejecutado en prácticamente cualquier navegador web moderno sin necesidad de plugins o complementos adicionales. Esto asegura que las

Tabla 55: Cuadro Comparativo de los Sistemas de Gestión de Bases de Datos No Relacionales

	Cassandra	MongoDB	Redis	Amazon DynamoDB
Tipo de base de datos	NoSQL <i>wide-column</i>	NoSQL Orientado a documentos	NoSQL clave-valor	NoSQL clave-valor
Licencia	Código abierto	Código abierto SSPL	Código abierto BSD 3-clause	Vendor
Cumplimiento ACID	No	Si	Si	Si
Lenguaje de consulta principal	CQL	JavaScript	Permite el uso de varios lenguajes	DQL
Principales casos de uso	Análisis social, análisis en tiempo real, venta al por menor y mensajería	Gestión de IoT, análisis en tiempo real, desarrollo de aplicaciones, inventario y personalización	Almacenamiento en caché, colas, filtrado y estadísticas	Juegos, comercio minorista, servicios financieros, publicidad y transmisión de medios
Seguridad	Seguridad integrada para la autorización, cifrado y autenticación, la seguridad está desactivada de forma predeterminada para facilitar su uso dentro de los clústeres	Seguridad incorporada para autorización, autenticación y encriptación	Se inicia automáticamente en modo de protección y ofrece sugerencias de seguridad	Seguridad integrada para datos y aplicaciones; software, hardware, instalaciones y red seguros del proveedor
Escalabilidad	Horizontal	Horizontal	Horizontal	Horizontal

aplicaciones frontend desarrolladas con JavaScript sean accesibles y funcionen de manera consistente en diferentes entornos y dispositivos.

▪ Procesamiento asíncrono

El procesamiento asíncrono es una de las características más útiles del lenguaje JavaScript. Usando JavaScript, un bloque del script no podrá detenerse o dejar que el otro bloque de código espere a que comience la respuesta. Si se está procesando una solicitud, otras también trabajarán en paralelo con la solicitud anterior en lugar de esperar la respuesta de la solicitud anterior. Ahorra mucho tiempo al ejecutar scripts en paralelo.

▪ Poca carga de Procesamiento del servidor

JavaScript permite realizar funcionalidades básicas en el lado del cliente. Por tanto el servidor no tendrá que procesar las funcionalidades básicas que mejoran el rendimiento del servidor.

6.4.4. Análisis y Elección del servidor de alojamiento

Para el presente proyecto se requiere una capacidad de almacenamiento que nos permita el uso de archivos multimedia, concretamente archivos de video.

Tipo de Multimedia	Ejemplos
Imágenes	JPEG, GIF, PNG, Archivos TIFF
Audio	MP3, WAV, ACC
Video	QuickTime, MP4, Youtube

Tabla 56: Tipos de Archivo Multimedia

Si se requiere alojar una aplicación web cuyo contenido predominante sea multimedia, se deberá de tomar en cuenta los siguientes factores:

- Ancho de banda

El ancho de banda y el almacenamiento son una función de su plan de alojamiento. Generalmente un plan de alojamiento en la nube permite pagar por un uso menor durante las horas de menor actividad y explotar cuando recibe mucho tráfico. No se paga por ancho de banda o espacio de almacenamiento que en realidad no se utiliza.

- Almacenamiento

Si se tiene una gran cantidad de archivos multimedia, entonces se requiere gran capacidad de almacenamiento. Lo ideal sería contar con almacenamiento ilimitado, pero esto significaría un aumento considerable al costo de alojamiento. Se debe de analizar un aproximado de almacenamiento necesario y en base a eso se deberá elegir el plan de almacenamiento que mejor cubra esas necesidades.

- CDN (*Content Delivery Network*)

Lo último que debe considerar con el alojamiento de medios es una red de entrega de contenido o CDN. Una CDN intenta mejorar el rendimiento de la transmisión de medios mediante la ubicación de servidores que están geográficamente cerca del usuario mediante sofisticados algoritmos de ubicación y de ruteo.

Sabiendo lo anterior, se procede a realizar el análisis de distintos proveedores de alojamiento con la propósito de seleccionar saber aquel que ofrezca la capacidad de alojar y almacenar contenido multimedia.

- AWS Amplify Hosting AWS Amplify Hosting es un servicio de alojamiento e Integración Continua/Entrega Continua completamente administrado para aplicaciones estáticas, rápidas, seguras, fiables, renderizadas del lado del servidor y que escalan a la par. Algunas de sus principales ventajas son:

- **Implementación de contenido web con rapidez**

AWS permite la implementación continua de una aplicación web estática o renderizada del lado del servidor, una página de inicio de la aplicación móvil o una aplicación progresiva en cada confirmación de código[45].

- **CDN**

Cuenta con la red de entrega de contenido (CDN) de Amazon CloudFront con cientos de puntos de presencia en todo el mundo. Lo que permite un tiempo de respuesta de hasta menos de 10ms.

- **Monitoreo**

AWS cuenta con un sistema de monitoreo de tráfico en tiempo real. También permite crear alarmas personalizadas para enviar notificaciones cuando la métrica haya superado el límite establecido.

- **Precio**

AWS cuenta con dos planes, uno gratuito y otro de pago, a continuación, se detallan las características de cada uno.

<p>Aloje una aplicación</p> <p>Gratis por 12 meses</p> <p>Implemente fácilmente su aplicación web o sitio web en la red de entrega de contenido (CDN) de AWS rápida, segura y confiable mediante Amplify Hosting.</p> <p>CREACIÓN E IMPLEMENTACIÓN</p> <p>Sin costo hasta los 1000 minutos de creación por mes</p> <p>ALMACENAMIENTO DE DATOS</p> <p>Sin costo hasta los 5 GB almacenados en la CDN por mes</p> <p>TRANSFERENCIA SALIENTE DE DATOS</p> <p>Sin costo hasta los 15 GB por mes</p>	<p>Aloje una aplicación</p> <p>Pago por uso</p> <p>¿Necesita más capacidad que el nivel gratuito? Amplify cambia a los precios de pago por uso una vez que supera los límites del nivel gratuito, sin interrupciones en sus aplicaciones.</p> <p>CREACIÓN E IMPLEMENTACIÓN</p> <p>0,01 USD por minuto</p> <p>ALMACENAMIENTO DE DATOS</p> <p>0,023 USD por GB por mes (este cargo se repite hasta que se elimine la aplicación)</p> <p>TRANSFERENCIA SALIENTE DE DATOS</p> <p>0,15 USD por GB servido</p>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figura 15: Planes de Alojamiento AWS Amplify

- **A2Hosting**

- **Alojamiento Compartido**

A2 Hosting cuenta con dos tipos de planes de alojamiento, el primero con costo de renovación que va desde \$10.99 dls . Si se llegara a necesitar un servidor dedicado, el plan que nos ofrece está característica tiene un precio de \$25.99 por mes.

- **VPS**

VPS significa servidor privado virtual por sus siglas en inglés. Es una forma de dividir un servidor en (sub)servidores individuales más pequeños. Esto significa que se puede configurar de acuerdo a las necesidades del cliente y no tener que compartir recursos con otros clientes. Un VPS en A2 Hosting cuesta desde tan solo \$7.65 al mes hasta alrededor de \$200 dependiendo del número de subdivisiones necesarias.

- **CDN** A2 Hosting no ofrece servicio de CDN, sin embargo, cualquiera de sus planes ofrecen la posibilidad de contratar un servicio de CDN proveedores externos pagando un precio extra del precio base dependiendo del plan contratado.

- **HosGator**

- **Tiempo de Respuesta**

De acuerdo con el sitio Bitcatcha, que se dedica a evaluar los tiempos de respuestas de los servidores de páginas web, HostGator posee tiempos de respuestas mínimos para el territorio de Estados Unidos. A continuación se muestra los tiempos de respuesta de distintos territorios:

US (W)	US (E)	London	Singapore	Sao Paulo
32 ms	36 ms	221 ms	221 ms	139 ms
Bangalore	Sydney	Japan	Canada	Germany
717 ms	205 ms	153 ms	36 ms	140 ms

Figura 16: Tiempos de Respuesta de servidores de HostGator

HostGator posee solo 2 datacenters, uno localizado en el área oeste, y otro en el este. Esto puede ocasionar que los tiempos de respuesta no sean los mejores en áreas que estén alejadas de las antes mencionadas

- **Disponibilidad** HosGator ofrece una disponibilidad del 99.9 %, esto quiere decir que los sitios que se decida alojar en los servidores de HostGator deberían ser accesibles la mayoría del tiempo.

Después de haber analizado los distintos proveedores de alojamiento así como sus características, se llegó a la conclusión de utilizar AWS Amplify. Ya que este contiene una suite de herramientas que facilitan la implementación de una aplicación web, como por ejemplo S3, que es un servicio de alojamiento en la nube utilizado para archivos multimedia de gran tamaño, además de contar con su propia CDN que facilitará el acceso a los datos almacenados.

7. Capítulo IV: Diseño

7.1. Diseño del Módulo Central de Procesamiento

Si todos los sistemas funcionan correctamente, el Módulo de Procesamiento Central entrará en modo de espera por los datos proporcionados por el Submódulo de Visión Artificial. En caso de que este último envíe una señal de alerta de somnolencia, el Módulo Central activará la alarma en forma de buzzer.

Posteriormente, se obtendrá la ubicación geográfica con la ayuda del Módulo de Comunicaciones. Se realizará un reporte de incidencia que contendrá la fecha, hora, ubicación y un pequeño video-clip del momento en que se detectó la somnolencia. Este será enviado a la Estación Base, que se encargará de almacenarlo en su respectiva base de datos. Como se indicó inicialmente, el sistema estará disponible mientras se encuentre conectado a una fuente de alimentación; esto significa que el estado del sistema se encuentra conectado.

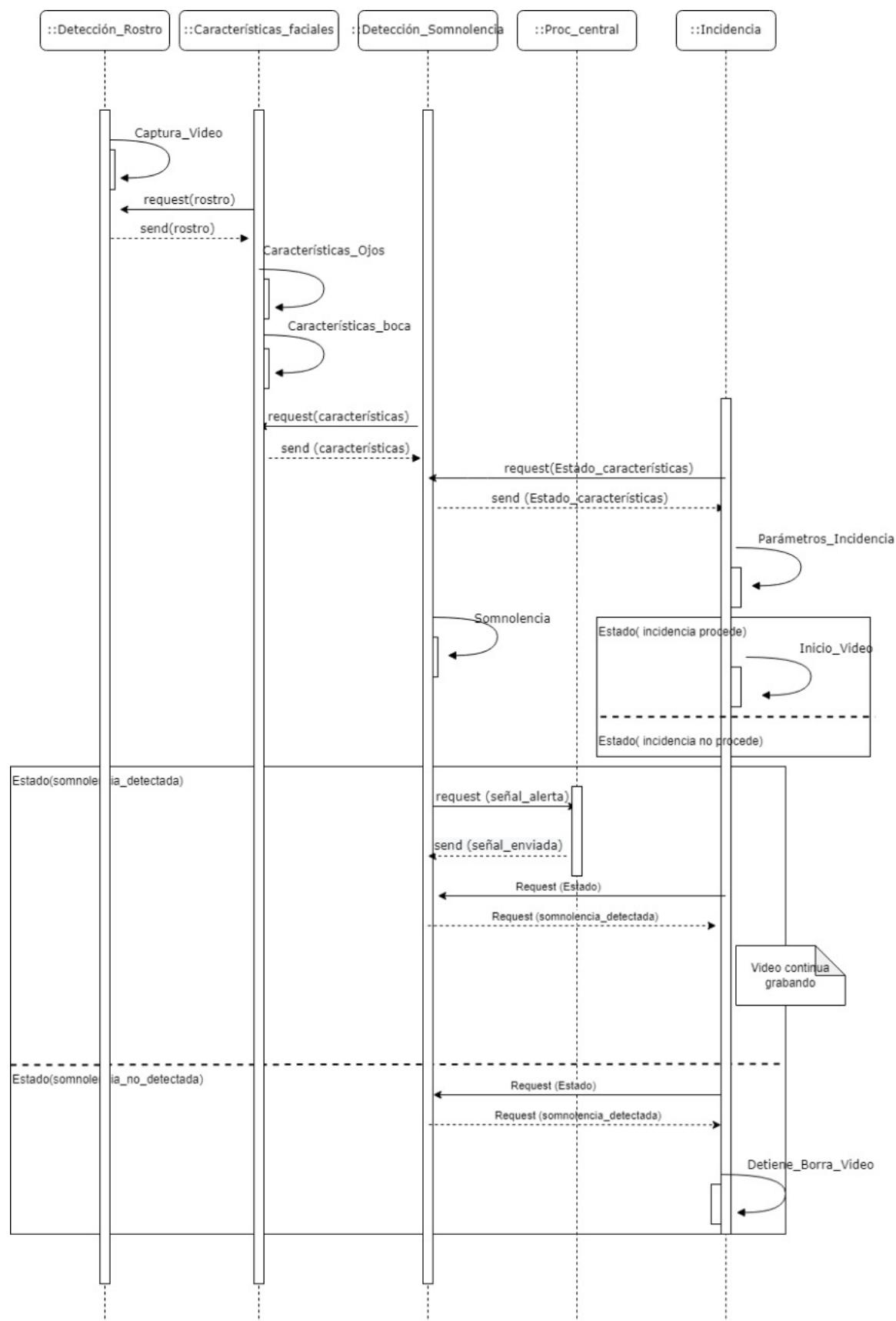


Figura 17: Diagrama de Secuencia - Módulo Central de Procesamiento

Siguiendo los procesos de la Figura 4, se procede a dar un análisis superficial en la concurrencia de los mismos. También se detallan las peticiones y respuestas de los distintos submódulos y sistemas.

7.1.1. Elección de la Unidad Contenedora de Procesamiento

Para el diseño de la unidad contenedora del módulo de procesamiento, se tomaron en cuenta los elementos físicos que estarán dentro de la unidad y sus respectivas medias. Cabe mencionar que los elementos que respectan al modelo del ordenador, el modelo de la cámara, el modelo del zumbador y el modelo de la microSD fueron previamente seleccionados en la secciones [6.2.1](#), [6.2.7](#), [6.2.2](#) y [6.2.3](#).

Elementos físicos que contendrá la unidad:

- **Nvidia Jetson Nano**

De acuerdo con las especificaciones físicas de la Jetson Nano se tiene las siguientes medidas en milímetros:

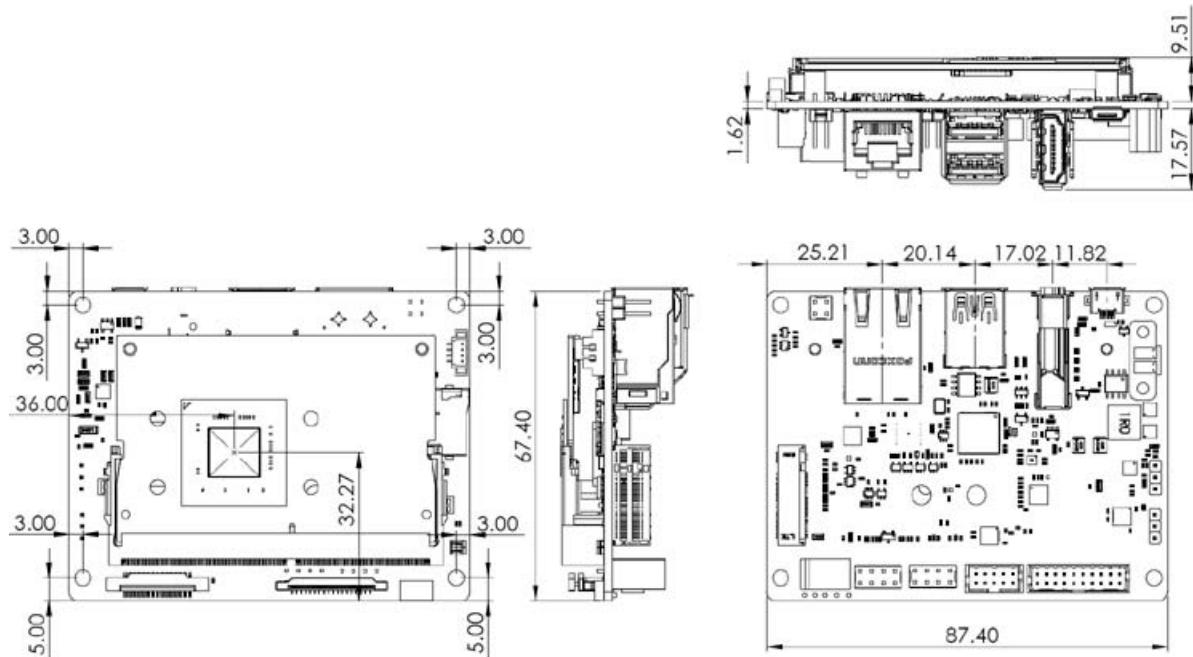


Figura 18: Dimensiones Nvidia Jetson Nano



Figura 19: Nvidia Jetson Nano

- Zumbador Pasivo KY-006
Dimensiones 18 x 15mm.



Figura 20: Zumbador Pasivo KY-006

- Modulo de Cámara IMX219-160
Dimensiones de la placa: 11 x 5.11 x 2.39 cm.



Figura 21: Modulo de Cámara IMX219-160

- Cable macho-hembra



Figura 22: Jumpers

Se utilizarán 2 cables macho-hembra para la conexión del zumbador pasivo hacia los pines GND y Vcc de la Jetson Nano. Largo 10 cm.

- Micro SD
Dimensiones: 15 x 11 x 1 mm.



Figura 23: Micro SD

En un principio se tenía propuesto diseñar la unidad contenedora de procesamiento. Sin embargo, al buscar alternativas, se encontró una carcasa específicamente diseñada para la Nvidia Jetson Nano, que cuenta con una plataforma para colocar una cámara digital además de orificios para antenas 4G, por lo que se decidió a utilizar esta carcasa para proteger los componentes.



Figura 24: Carcasa Waveshare

Esta carcasa está hecha de metal y además incluye un ventilador para disipar el calor.



Figura 25: Componentes de la Unidad Contenedora de Procesamiento

7.1.2. Diseño del Submódulo de Visión Artificial

Con base a los requerimientos definidos en la sección de Análisis, se procedió a realizar distintos diagramas que muestren las interacciones y el funcionamiento de los distintos módulos.

En el siguiente Diagrama de Actividades se describen las actividades y sus interacciones con el Submódulo de Visión Artificial y la Estación Base. Para que el sistema principal pueda iniciar, se necesita que el conductor encienda el auto, ya que este sistema funcionará utilizando la alimentación eléctrica.

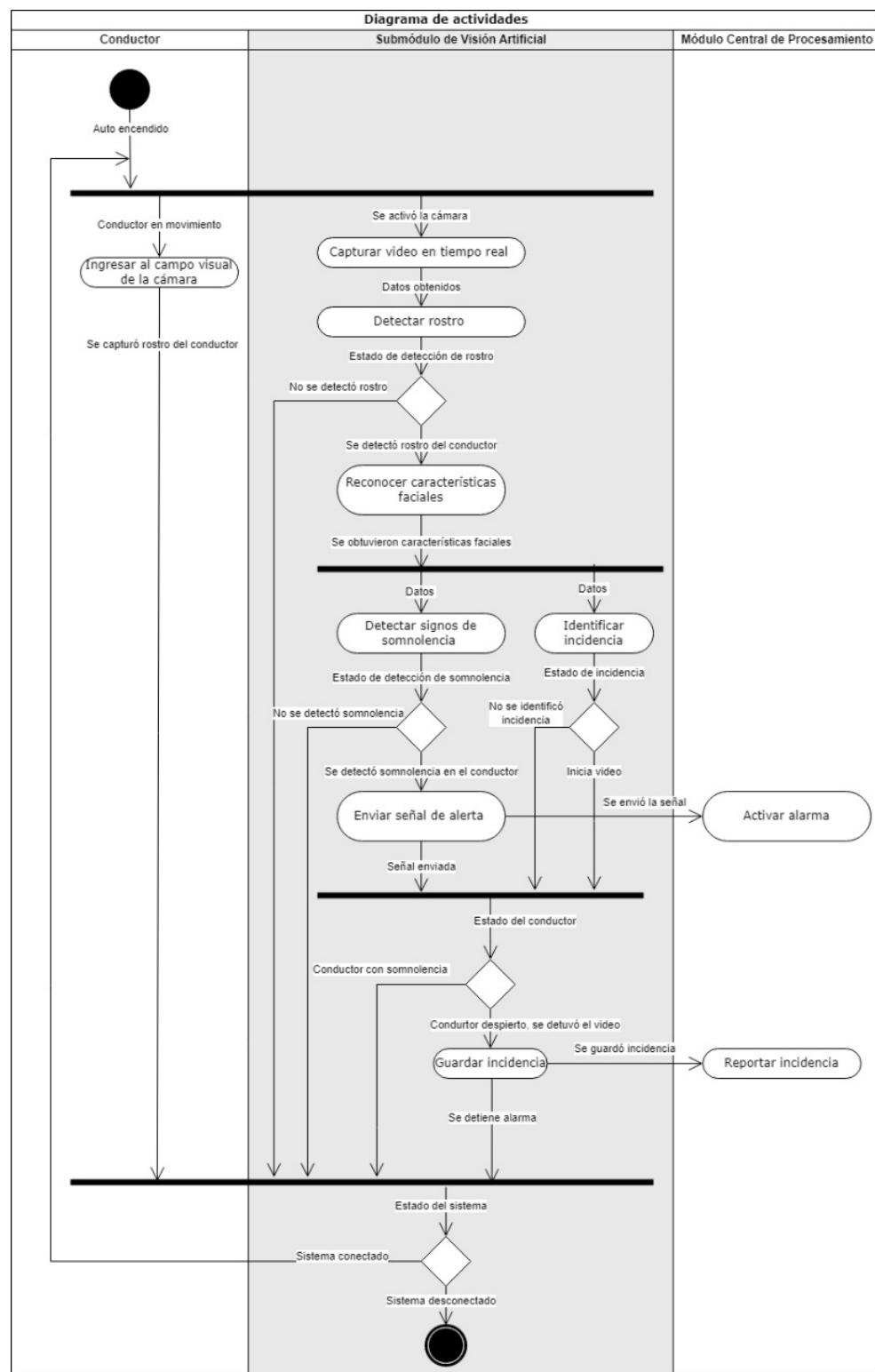


Figura 26: Diagrama de Actividades - Submódulo de Visión Artificial

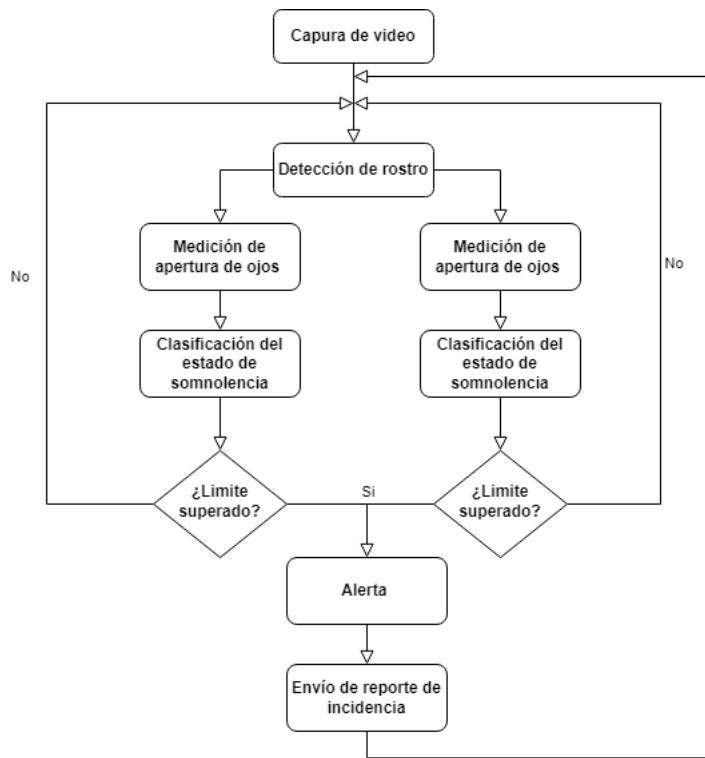


Figura 27: Diagrama de Flujo - Submódulo de Visión Artificial

La cámara digital capturará la imagen en tiempo real a una resolución de 640x480 píxeles. Dado que buscamos ahorrar recursos de procesamiento, el uso de una mayor resolución o una tasa de cuadros por segundo más alta podría resultar en un funcionamiento ineficiente de los procesos en los distintos módulos.

Para la detección de rostro, ojos y boca, se utilizarán las librerías Dlib y OpenCV. Posteriormente, se determinará si el conductor tiene los ojos abiertos o cerrados mediante la medición de distancias utilizando puntos de referencia. En caso de detectar ojos cerrados, se iniciará un contador. Si este contador llega a los 4 segundos de detectar ojos cerrados, el sistema determinará que el conductor presenta un caso de somnolencia y activará la alarma para alertar al conductor. Posteriormente, se generará un reporte de incidencia que incluirá datos del conductor y de la incidencia, como fecha, hora y ubicación, así como un videoclip de menos de 1 minuto que muestre el momento en que se detectó la incidencia. El sistema continuará activando la alerta hasta que se detecte un cambio en el estado de los ojos (de cerrados a abiertos). Finalmente, el reporte de incidencia será enviado a la Estación Base para su posterior revisión.

- **Puntos de Referencia**

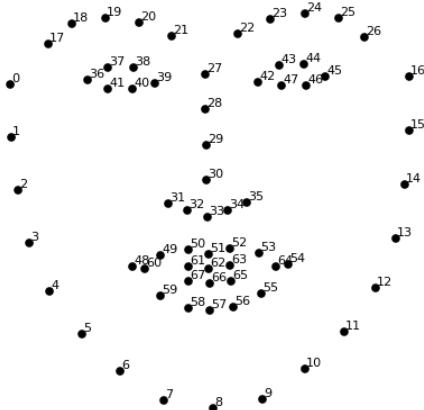


Figura 28: Puntos de referencia

A su vez, se utilizarán puntos de referencia con la ayuda de el software OpenCV y el modelo iBUG 300-W para delimitar la región de interés de la boca.

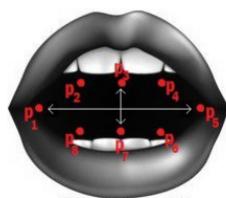
En la siguiente tabla se muestra la delimitación y agrupación de los puntos de interés, tales como: ojo derecho, ojo izquierdo y boca. De este modelo de detección de rostro, solo serán utilizados el grupo de 48 a 67, que delimitarán la región de la boca.

Partes	Puntos de Referencia
Boca	[48-67]
Ojo Derecho	[36-41]
Ojo Izquierdo	[42-46]

Tabla 57: Agrupación de Puntos de Referencia

■ Mouth Opening Ratio

Debido a que bostezar es un signo de cansancio, se propone medir el tamaño y la forma de la boca es necesario para identificar un bostezo. Para esto, se utilizará el *Mouth Opening Ratio* que es un método que utiliza puntos de referencia para medir la apertura de la boca. Entre más grande sea este valor, más es la aperatura de la boca, por lo tanto cumple con las características de un bostezo.

Figura 29: *Mouth Opening Ratio*

La formula general para calular el *MOR* es la siguiente:

$$MOR = \frac{\|p_2 - p_8\| + \|p_3 - p_7\| + \|p_4 - p_6\|}{2 \|p_1 - p_5\|} \quad (2)$$

Utilizando nuestros puntos de referencia, se podrá calcular el MOR de la siguiente manera:

$$MOR = \frac{\|P49 - P59\| + \|P51 - P57\| + \|P53 - P55\|}{2 \|P48 - P54\|} \quad (3)$$

De tal manera, cuanto mayor sea el valor del *MOR*, mayor será la apertura de la boca. Por lo cuál, se tomará un valor de 0.60, que es un estándar establecido para el valor de un bostezo.[?]

7.2. Diseño del Módulo de Comunicaciones

Para el diseño de este módulo se tiene contemplado ejecutar dos procesos asíncronos, es decir, que se ejecutan al mismo tiempo o de manera paralela.

A continuación se muestra el diagrama de Flujo del Módulo de Comunicaciones

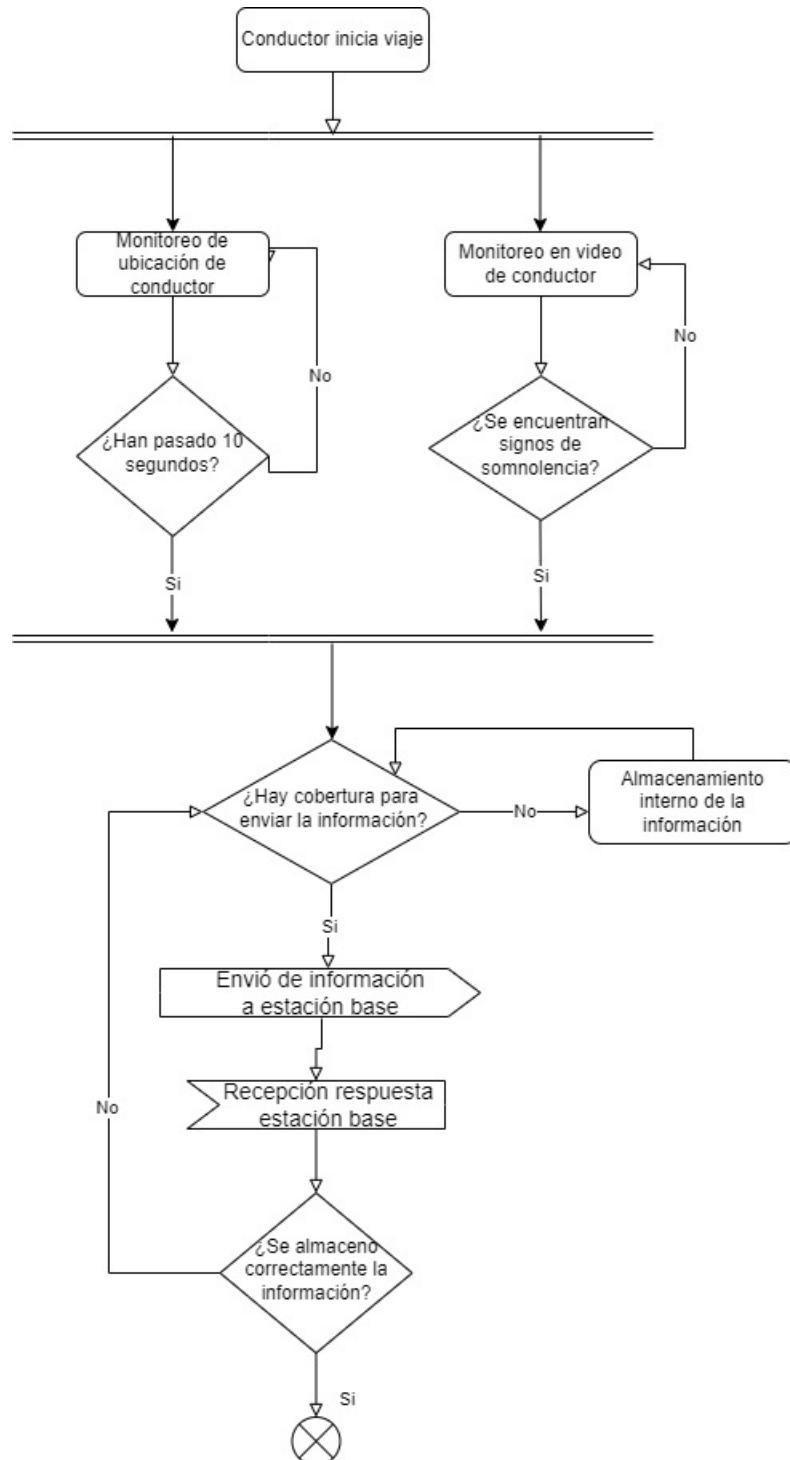


Figura 30: Diagrama de Flujo del Módulo de Comunicaciones

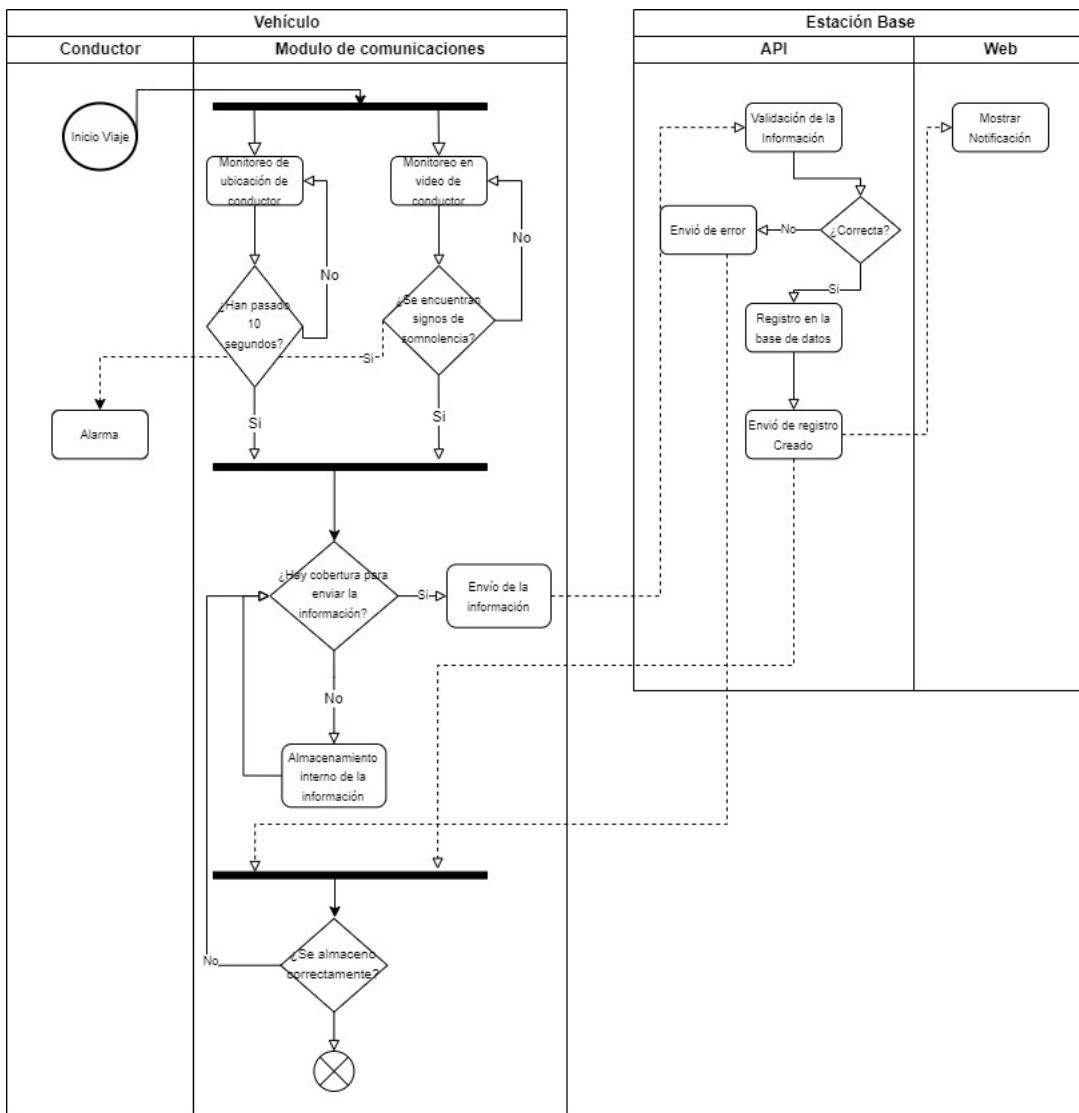


Figura 31: Diagrama Actividades del Módulo de Comunicaciones

a ubicación del conductor se obtendrá mediante el Módulo de Telemetría cada 10 segundos, con la intención de no consumir demasiados recursos de procesamiento y energía. Posteriormente, esta información será enviada a la Estación Base utilizando el módulo SIM27600G. En caso de no contar con cobertura que permita la conexión a la estación base, la ubicación del conductor permanecerá en la última obtenida hasta que se actualice desde la Unidad de Procesamiento Central.

7.3. Diseño de la Estación Base

Con base en los requerimientos, se plantea realizar el diseño de la arquitectura del Módulo de Estación Base, integrando el Módulo de Procesamiento Central. El cual hará énfasis en la organización y comunicación de los elementos que lo conforman.

La Figura 32 a muestra la arquitectura del sistema de Estación base, el cual se compone de las interacciones de los elementos que permitirán al usuario acceder e interactuar con la aplicación web. El diagrama también muestra la participación del Módulo de Procesamiento Central ya que el registro de incidencias y el envío del video de incidencia serán realizados por dicho módulo. La información de los reportes de incidencias y el registro de los conductores serán almacenados en DynamoDB, mientras que los videos de incidencia se almacenarán en Amazon S3. Las credenciales de los Usuarios Administradores que tendrán permitido acceder a la aplicación se almacenarán en Amazon Cognito. Para la aplicación web, el *backend* será desarrollado en node.js mientras que el *frontend* será desarrollado en React, posteriormente los archivos se almacenarán en un repositorio dentro de Github. La aplicación web será alojada y desplegada desde AWS Amplify, lo cual le permitirá al cliente acceder a la aplicación desde una URL.

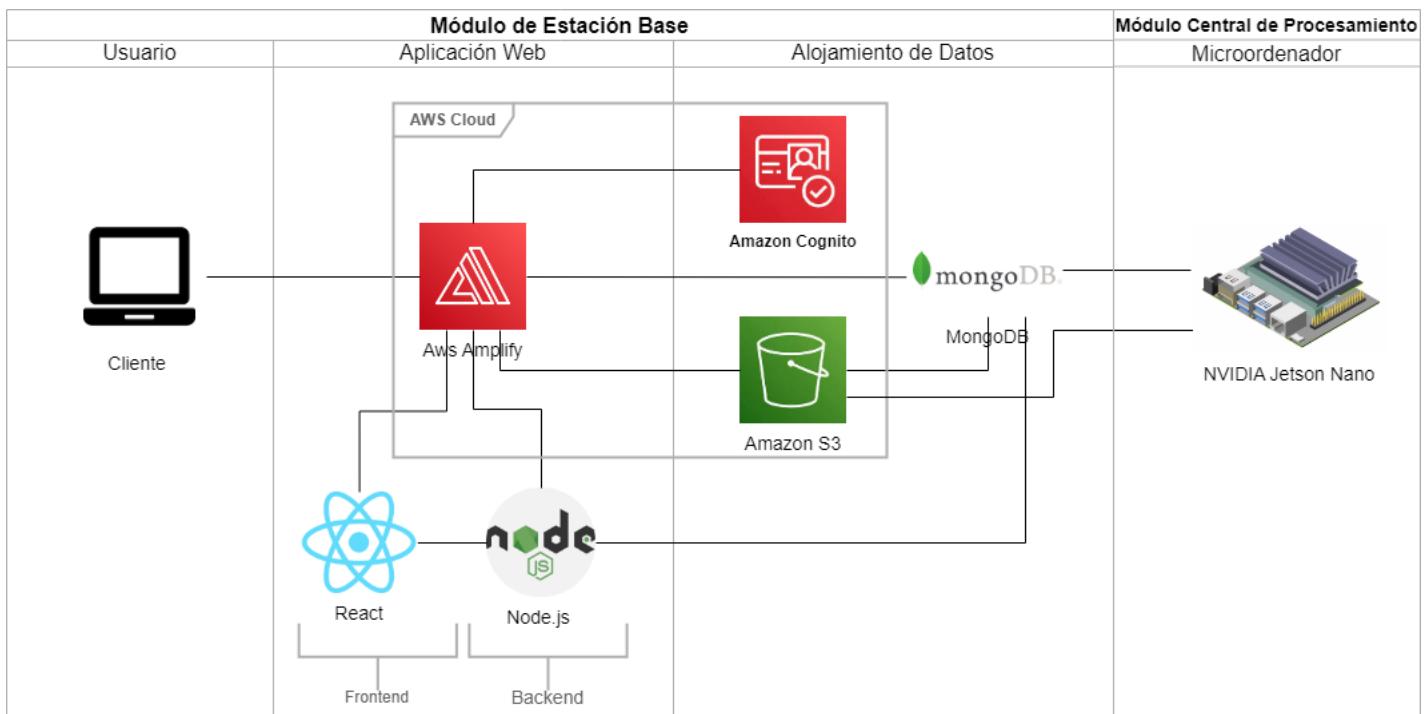


Figura 32: Diagrama de Diseño - Módulo de Estación Base

El diagrama de comunicación muestra las interacciones entre elementos que se involucran en cada requerimiento funcional. Así mismo, dan profundidad al diagrama de Arquitectura del Módulo de Estación Base. La aplicación web hará uso de la Base de Datos y el *backend* de la aplicación web, que se realizará con Node.js. Mientras que para el despliegue de la aplicación web se requiere del servidor AWS Amplify para permitirle al usuario entrar a la aplicación web. El sistema almacenará el video de incidencia en Amazon S3, enviado desde el Módulo Central de Procesamiento, al cual se podrá acceder por medio del ID. El envío de incidencia requiere la interacción del Módulo Central

de Procesamiento el cual se encargará de realizar y enviar el Reporte de Incidencia, posteriormente será almacenado en la Base de Datos.

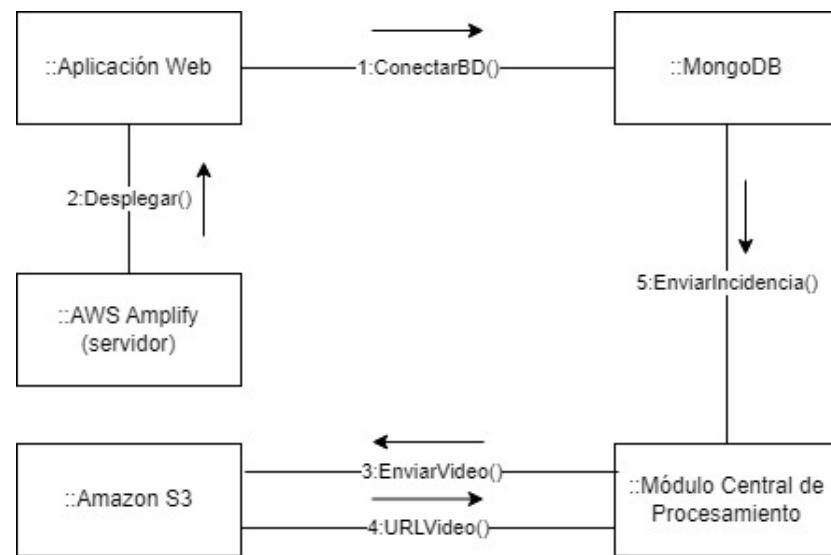


Figura 33: Diagrama de Comunicación - Módulo de Estación Base

7.3.1. Diseño de la Aplicación Web

Para el diseño y desarrollo de la Aplicación web, se decidió utilizar el lenguaje Javascript, ya que este cuenta con las herramientas y/o frameworks que mejor se adaptan a los requisitos previamente definidos. A su vez, se decidió utilizar el NodeJs para el sistema backend.

Node.js, es un entorno en tiempo de ejecución multiplataforma basado en JavaScript. Este es controlado por eventos, permitiendo establecer y gestionar múltiples conexiones al mismo tiempo. Gracias a esta característica, el bloqueo de procesos no existe. NodeJs trabaja fundamentalmente bajo dos características: *asincronía*, que permite la ejecución de varios procesos al mismo tiempo, y *Entrada/salida sin bloqueo*, que significa poder trabajar con múltiples solicitudes sin bloquear un hilo para una sola solicitud. [46]

NodeJs es capaz de manejar distintas peticiones sin que tener que esperar a que una petición sea respondida para continuar con la siguiente petición. De ahí la elección de este entorno de Javascript para el desarrollo de la aplicación web.

A continuación se muestran diversos diagramas de secuencias que describen los procesos de los requerimientos definidos en la sección de Análisis.

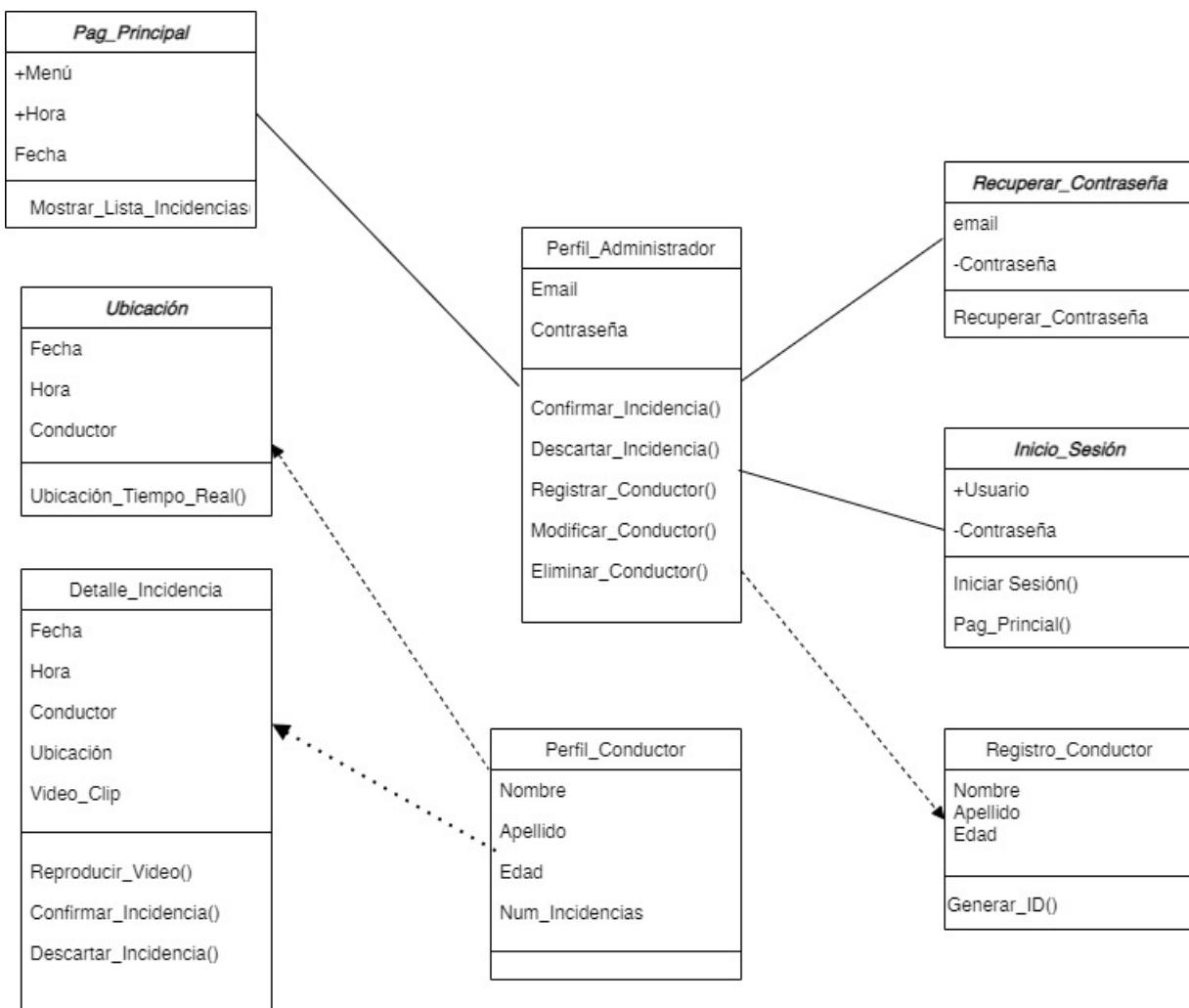


Figura 34: Diagrama de Clases - Aplicación Web

En la Figura 34 se pueden observar las clases que estarán dentro de la aplicación web, así como sus atributos y las interacciones entre estas.

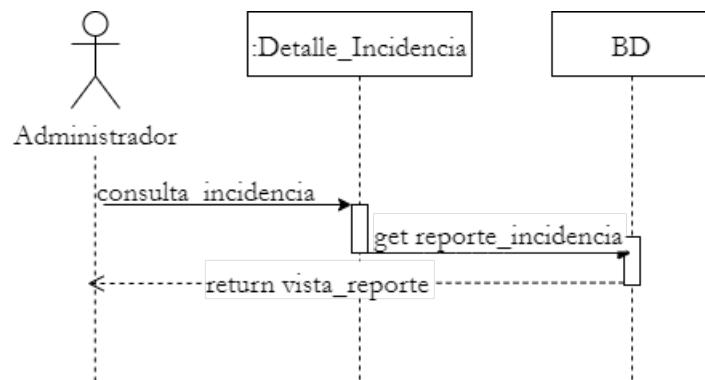


Figura 35: Diagrama de Secuencia Detalle Reporte Incidencia

El Administrador podrá consultar el reporte de incidencia de cada uno de los conductores. Accediendo a la base de de datos.

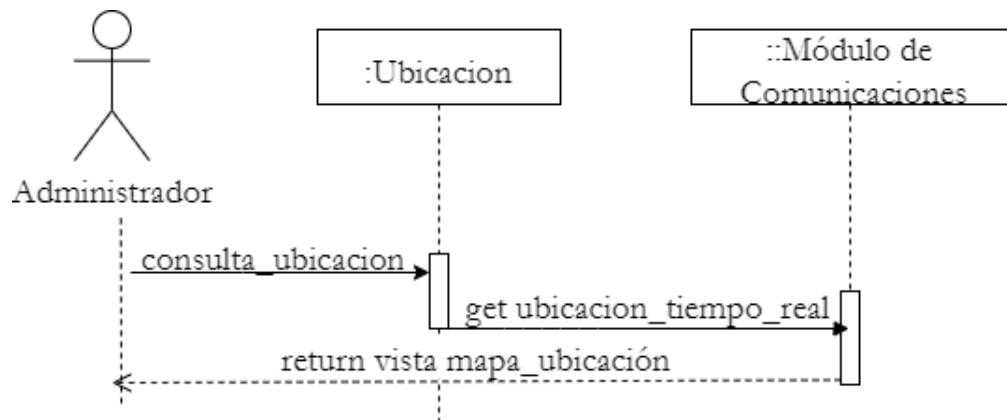


Figura 36: Diagrama de Secuencia Consultar Ubicación

De igual manera, el Administrador podrá consultar la ubicación en tiempo real del conductor.

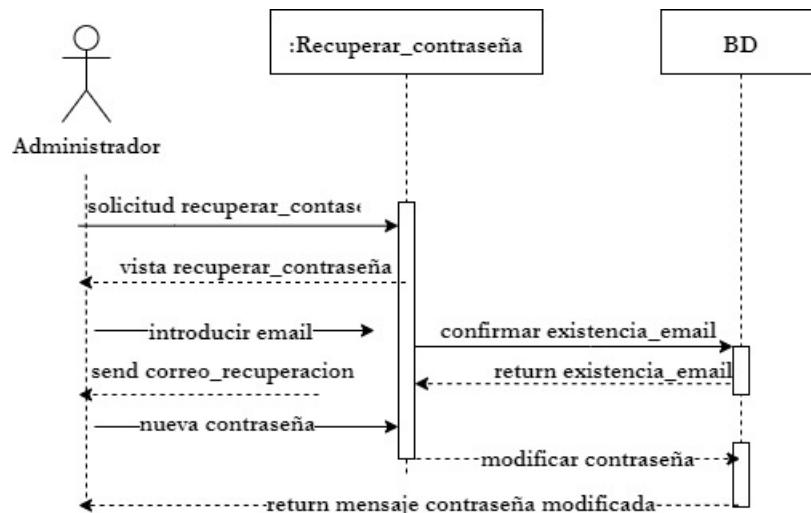


Figura 37: Diagrama de Secuencia Recuperar Contraseña

También podrá recuperar su contraseña en caso de que esta sea olvidada.

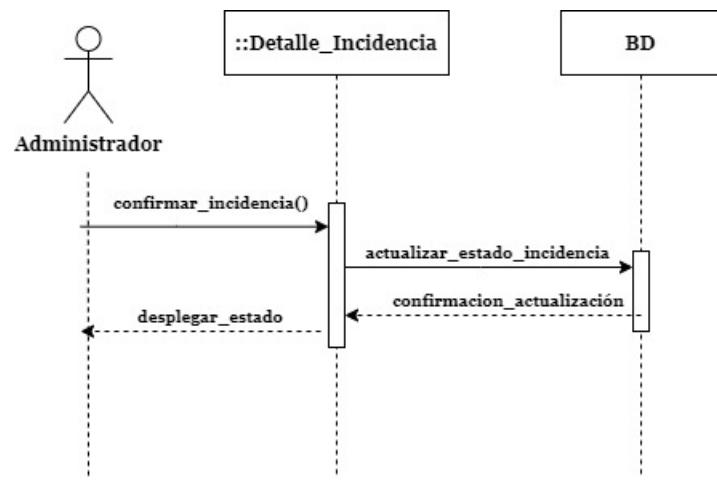


Figura 38: Diagrama de Secuencia Confirmar Incidencia

Para evitar falsos positivos, el Administrador podrá confirmar una incidencia una vez mirando el videoclip del incidente.

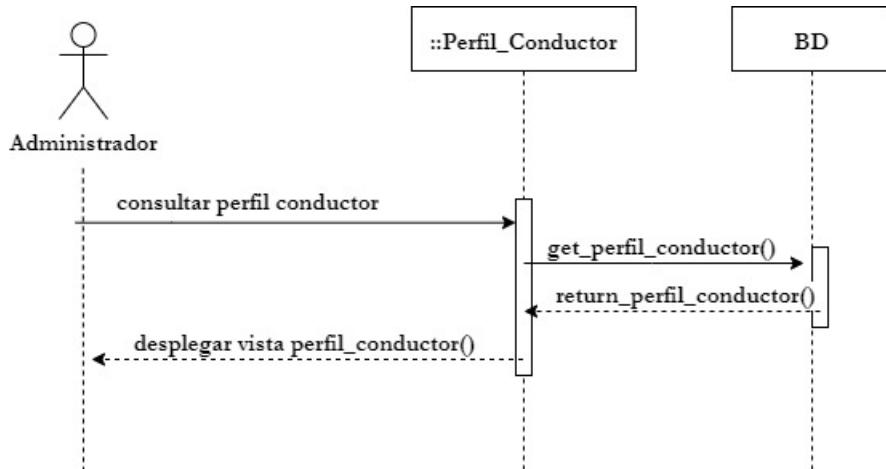


Figura 39: Diagrama de Secuencia Consultar Perfil

El Administrador podrá consultar el perfil de cada conductor.

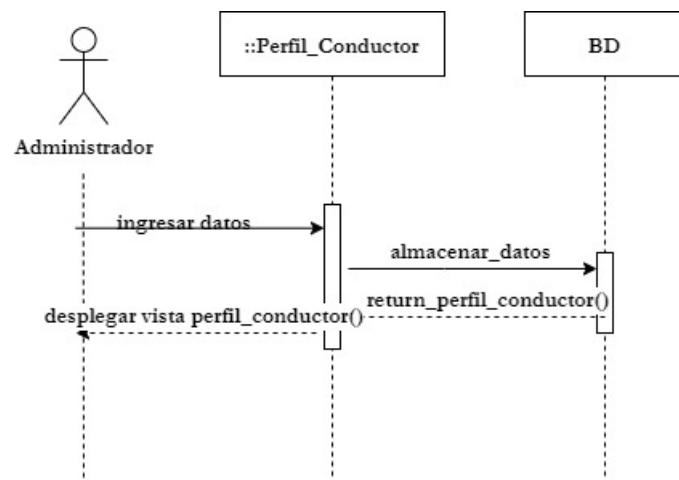


Figura 40: Diagrama de Secuencia Registrar Conductor

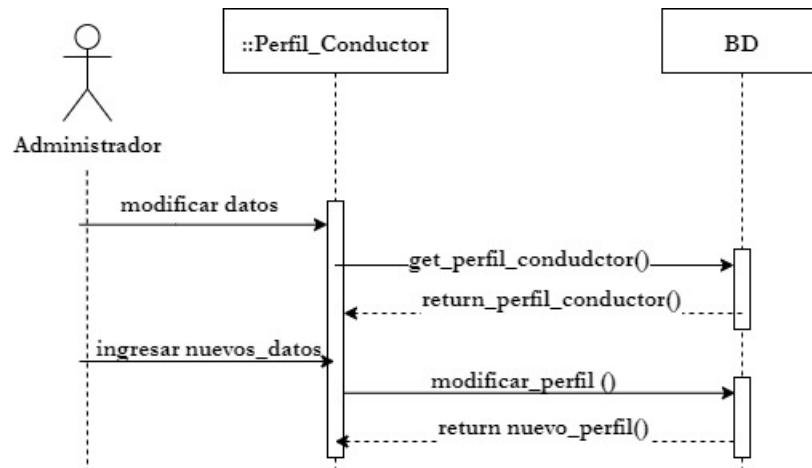


Figura 41: Diagrama de Secuencia Modificar Conductor

En caso de que los datos del conductor sean incorrectos, el administrador podrá modificarlos.

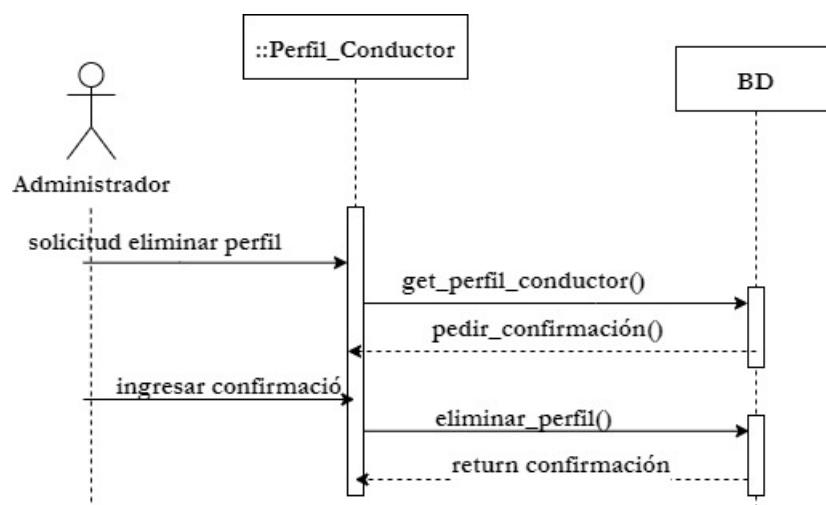


Figura 42: Diagrama de Secuencia Eliminar Conductor

El Administrador también tendrá la opción de dar de baja a un conductor.

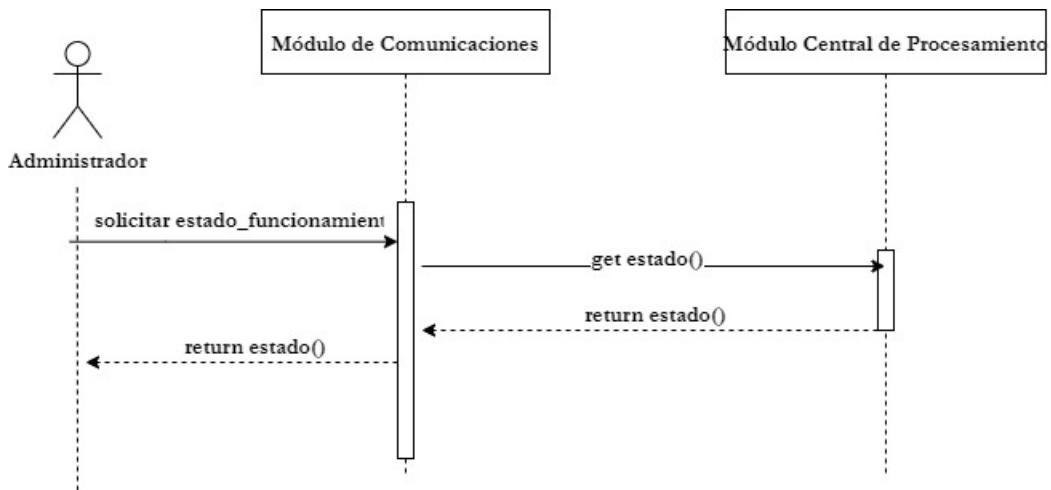


Figura 43: Diagrama de Secuencia Estado de los Periféricos

La aplicación web indicará si alguno de los periféricos no se encuentra en correcto funcionamiento, lanzando una alerta indicando el número de Unidad y a qué conductor pertenece.

A continuación se muestra el diseño propuesto de cada una de las vistas que tendrá la aplicación web.

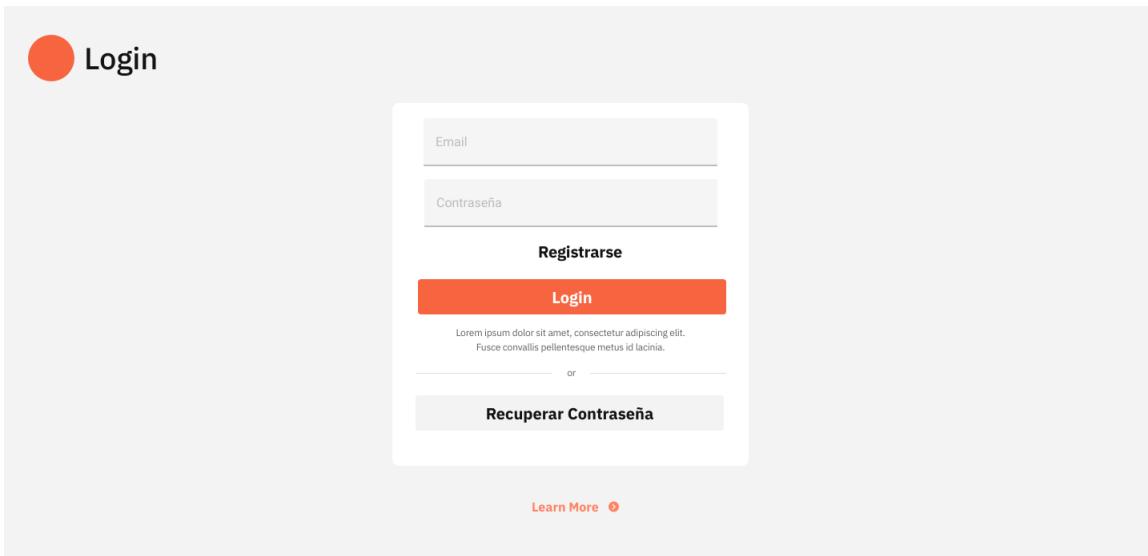


Figura 44: Página Inicio de Sesión

En la Figura 44 se observa la página donde el Administrador podrá iniciar sesión ingresando su correo y su contraseña.

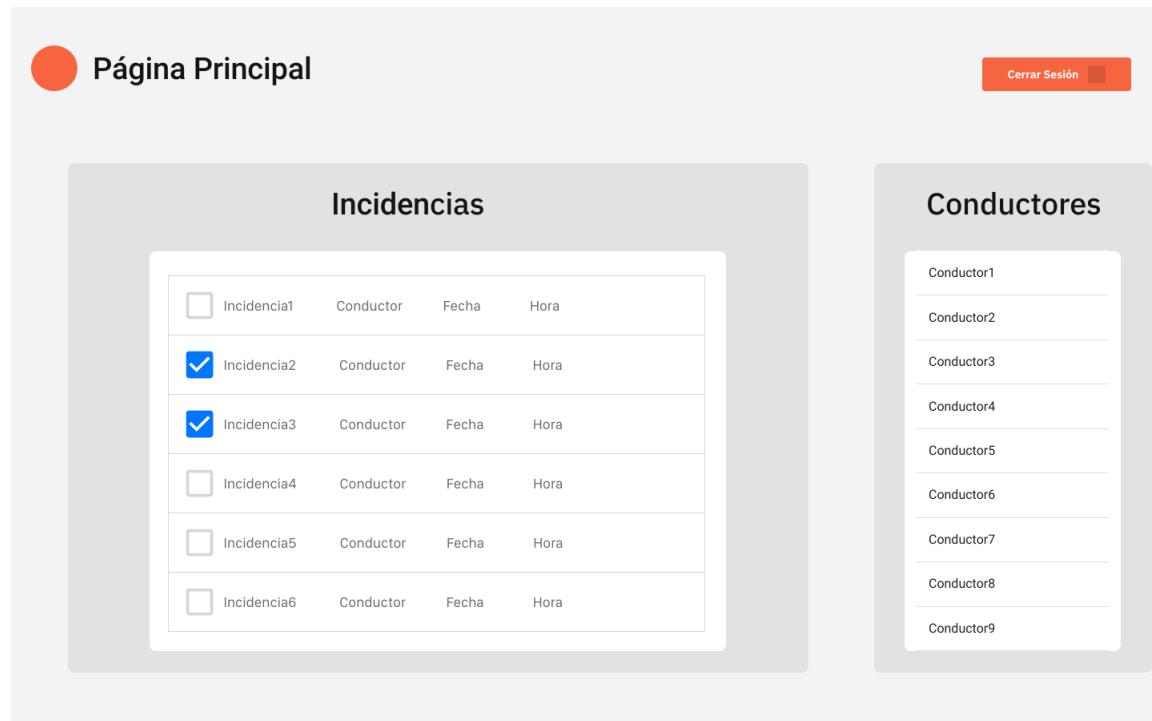


Figura 45: Página Principal

En la Figura 45 se muestra la página principal, dónde se muestran la lista de las incidencias, dónde cada una tendrá un estado de confirmada, rechazada, o pendiente de revisión. También se mostrarán una lista de todos los conductores para poder acceder al perfil de cada uno.

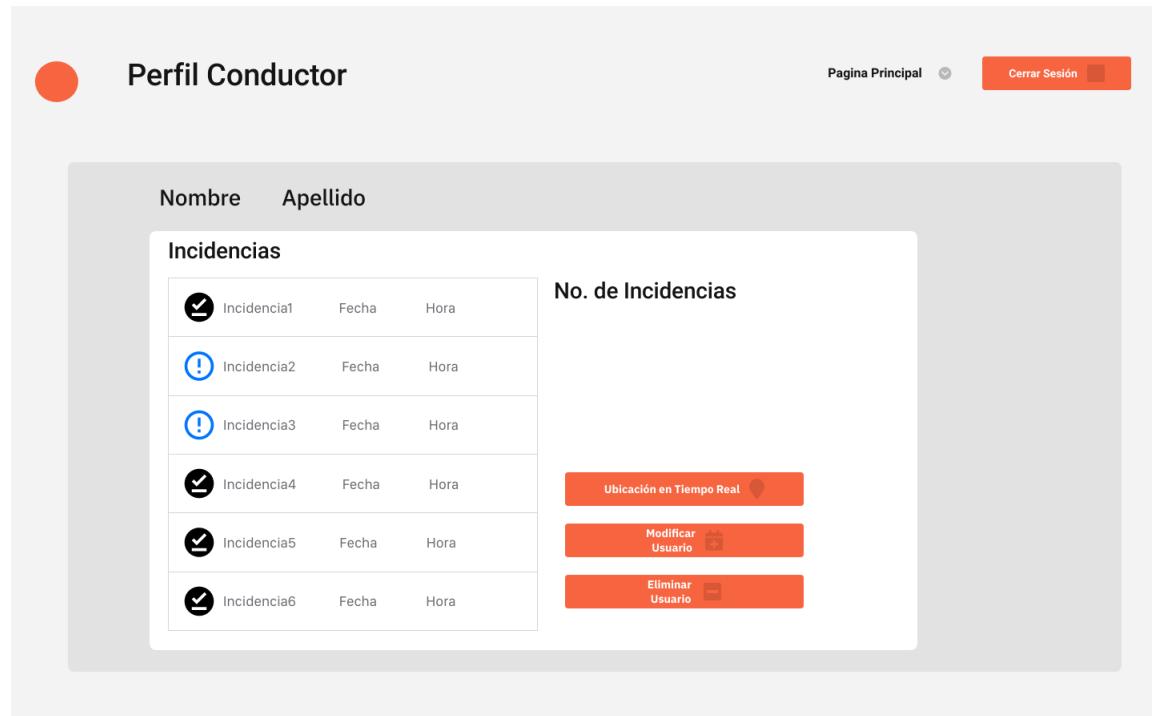


Figura 46: Página Perfil del Conductor

En la Figura 46 se aprecia la página de perfil del conductor, que mostrará sus datos personales, así como una lista de todas las incidencias acumuladas.

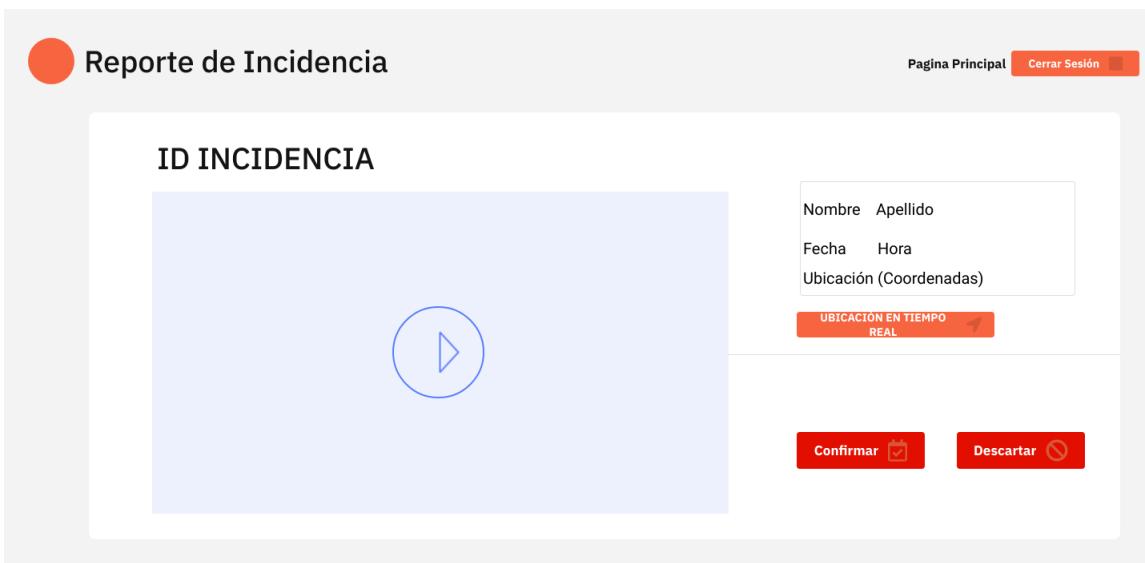


Figura 47: Página Detalle de Incidencia

En la Figura 47, se puede observar la página desde dónde el administrador podrá revisar el videoclip de las incidencias de cada conductor. También cuenta con las opciones de confirmar o rechazar la incidencia dependiendo de cuál sea el caso.



Figura 48: Página Ubicación en Tiempo Real

En la Figura 48 se muestra la página web donde el Administrador podrá consultar la ubicación de un conductor en tiempo real.

7.3.2. Diseño de la Base de Datos

DynamoDB es un servicio de base de datos NoSQL proporcionado por Amazon Web Services (AWS). Su modelo de datos difiere del modelo relacional tradicional y se basa en conceptos clave como tablas, índices y claves primarias.

1. Tablas:

- DynamoDB almacena datos en tablas. Cada tabla es un conjunto de ítems (datos) relacionados.
- Las tablas no tienen un esquema fijo; cada ítem en una tabla puede tener diferentes atributos.

2. Atributos:

- Cada ítem en una tabla es un conjunto de atributos. Los atributos son pares clave-valor.
- Los atributos pueden ser de diferentes tipos de datos, como texto, número, binario, lista y mapa.

3. Claves Primarias:

- Cada tabla en DynamoDB debe tener una clave primaria, que puede ser simple o compuesta.
- La clave primaria puede consistir en uno o dos atributos: la clave de partición y, opcionalmente, la clave de ordenación.
- La clave de partición determina la ubicación física de los datos en DynamoDB, mientras que la clave de ordenación ordena los ítems con la misma clave de partición.

4. Índices:

- DynamoDB admite índices locales y globales.
- Los índices locales están limitados a una única clave de partición y pueden tener una o varias claves de ordenación.
- Los índices globales pueden tener cualquier clave, ya sea solo de partición o compuesta de partición y ordenación.

```
{  
  "TableName": "Conductores",  
  "KeySchema": [  
    { "AttributeName": "IDConductor", "KeyType": "HASH" }  
  ],  
  "AttributeDefinitions": [
```

```
{ "AttributeName": "IDConductor", "AttributeType": "N" },
{ "AttributeName": "Nombre", "AttributeType": "S" },
{ "AttributeName": "Apellido", "AttributeType": "S" },
{ "AttributeName": "Incidencias", "AttributeType": "N" },
{ "AttributeName": "ContadorEnMarcha", "AttributeType": "N" }
],
"ProvisionedThroughput": {
    "ReadCapacityUnits": 5,
    "WriteCapacityUnits": 5
}
}
```

El modelo de la tabla para Conductores, se usará para registrar el nombre y apellido de cada conductor, así mismo se contabilizará el número de incidencias que presentó cada uno de ellos, el contador del conductor en marcha aumentará cada vez que el Módulo Central de Procesamiento envíe un reporte de incidencia a la base de datos. Si el Usuario Administrador del Módulo de la Estación Base revisa el video de la incidencia y lo cataloga como Descartar entonces se restará la incidencia, si esta es catalogada como Confirmar, el contador permanecerá igual. Es importante mencionar que el registro de cada conductor se realizará desde la Aplicación Web.

```
{
    "TableName": "Incidencias",
    "KeySchema": [
        { "AttributeName": "IdIncidencia", "KeyType": "HASH" }
    ],
    "AttributeDefinitions": [
        { "AttributeName": "IdIncidencia", "AttributeType": "N" },
        { "AttributeName": "IdConductor", "AttributeType": "N" },
        { "AttributeName": "FechaHora", "AttributeType": "S" },
        { "AttributeName": "NombreConductor", "AttributeType": "S" },
        { "AttributeName": "ApellidoConductor", "AttributeType": "S" },
        { "AttributeName": "EstadoIncidencia", "AttributeType": "S" }
    ],
    "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
    }
}
```

El modelo de la tabla de Incidencias, se usará para registrar el Reporte de Incidencia que presente un conductor, el cual será enviado desde el Módulo Central de Procesamiento, por tanto, se realizará una consulta previa a la colección Conductores para obtener el id, nombre y apellido del conductor en marcha. Posteriormente se realizará el reporte de la incidencia, el cual contendrá el Id del conductor, la fecha y hora de la incidencia, el nombre y apellido del conductor, y el estado de incidencia, este último muestra si la incidencia fue catalogada como descartada o confirmada por parte del Usuario Administrador después de revisar el video de la incidencia.

Amazon Cognito

Es una suite de herramientas que ofrece autenticación, autorización y administración de usuarios para aplicaciones móviles o web. Amazon Cognito utiliza dos componentes principales: los grupos de usuarios y grupos de identidades. Los grupos de usuarios se tratan de directorios que proporcionan a los usuarios de las aplicaciones opciones para inscribirse e iniciar sesión. Por otro lado, los grupos de identidades conceden a los usuarios acceso a otros servicios de *Amazon Web Services*. A continuación se muestra el funcionamiento de Amazon Cognito.

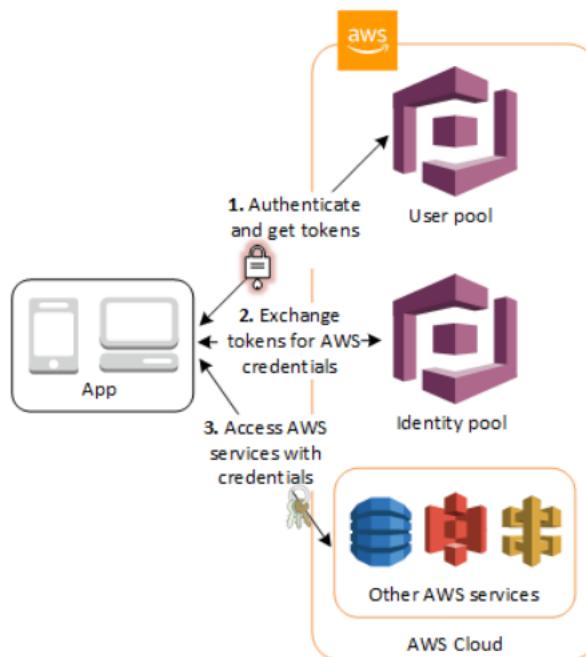


Figura 49: Arquitectura Amazon Cognito

1. En primer lugar, el usuario inicia sesión a través de su respectivo grupo de usuarios y recibe *tokens* de grupos de usuario después de una autenticación correcta.
2. Después, la aplicación intercambia dichos tokens del grupo de usuarios por las credenciales de AWS mediante un grupo de identidades.
3. Finalmente, el usuario puede utilizar estas credenciales de AWS para obtener acceso a otros servicios como Amazon S3.

Amazon Cognito se encuentra disponible en varias regiones alrededor del mundo.

Amazon S3 (*Simple Storage Service*)

Amazon S3, es un servicio de almacenamiento de objetos en la nube. Es utilizado para almacenar datos en la nube de una forma segura, eficiente y escalable. Este servicio utiliza elementos llamados *buckets*, que se encargan de almacenar objetos. Un objeto es un archivo o cualquier metadato que describa dicho archivo. Para almacenar datos en S3, primero se debe especificar el nombre de un bucket y la región donde se planea que opere la aplicación. Esto con la intención de que el acceso a los datos se realice de manera eficiente.

8. Implementación

8.1. Módulo Central de Procesamiento

8.1.1. Instalación del entorno de desarrollo en la NVIDIA Jetson Nano

El Jetson Nano Developer Kit utiliza una tarjeta microSD como dispositivo de arranque y almacenamiento principal. Por tanto, fue necesario instalar un entorno de desarrollo en la propia placa, para lo cual se requirió una tarjeta microSD de un mínimo recomendado de 32 GB, según la documentación de Nvidia [?].

Como primer paso, se descargó el *JetPack SDK 4.6.3* [47], que es un archivo de imagen del sistema operativo específicamente diseñado y optimizado para ser utilizado en la placa de desarrollo NVIDIA Jetson Nano. La imagen incluye un sistema operativo Linux, controladores de hardware específicos para la Jetson Nano y una configuración predefinida. Para el presente proyecto, se utilizó la última versión disponible para la Jetson Nano 4GB, JetPack SDK 4.6.3 [47].

Posteriormente, se instaló la imagen en la tarjeta microSD con el apoyo de BalenaEtcher 1.7.9, que se utilizó para crear una unidad de arranque.

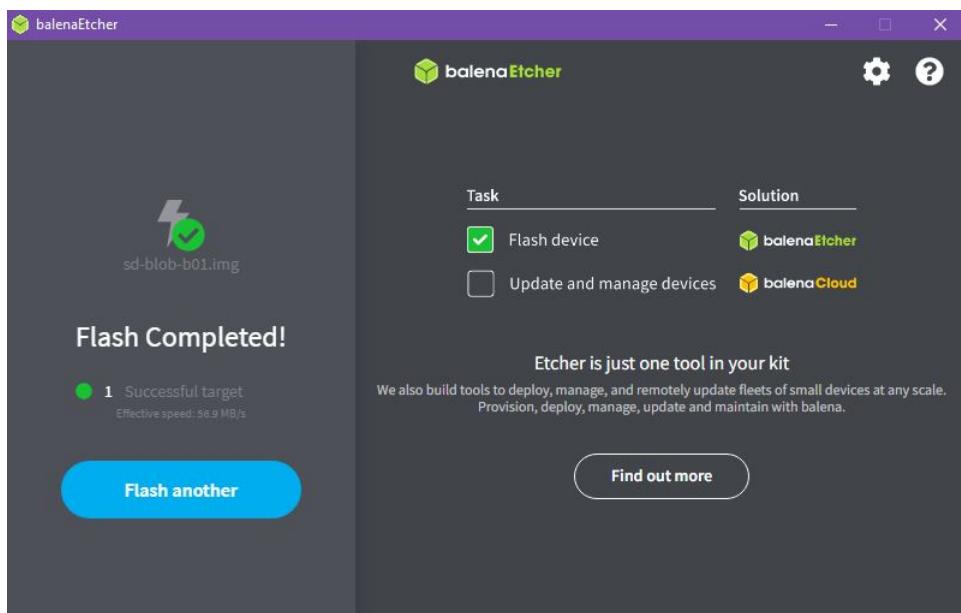


Figura 50: Instalación del JetPack SDK realizado con BalenaEtcher.

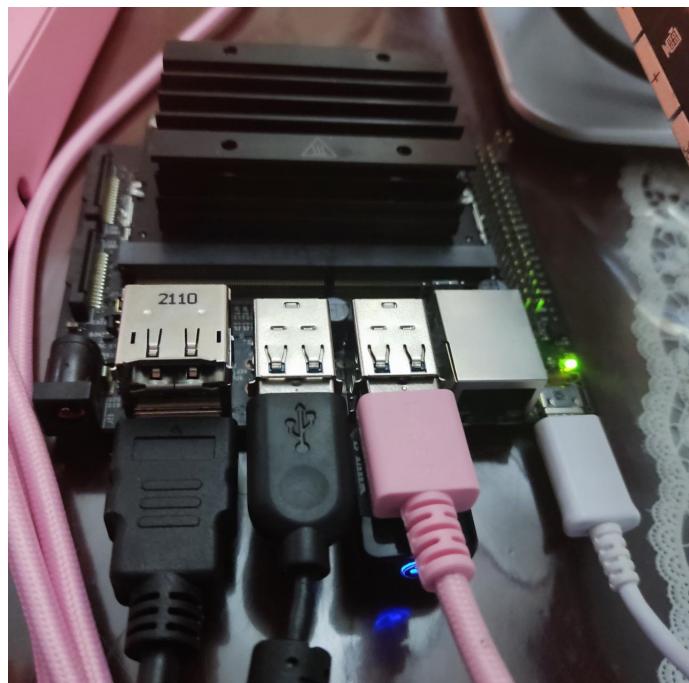


Figura 51: Periféricos de entrada para el Kit de desarrollo Jetson Nano.

Configuración y primer arranque

Para iniciar e interactuar con el kit de desarrollo, se requirió conectar un mouse, pantalla, teclado y una fuente de alimentación. Cabe mencionar que estos elementos no se incluyen con la compra del kit de desarrollo Jetson Nano. Posteriormente, se realizó la configuración inicial del sistema operativo, la cual incluyó los siguientes pasos:

- Revisar y aceptar el EULA del software NVIDIA Jetson.
- Seleccionar el idioma del sistema, la distribución del teclado y la zona horaria.
- Crear un nombre de usuario, contraseña y nombre de la computadora.
- Seleccionar el tamaño de partición de la aplicación; se utilizó el tamaño máximo sugerido.

Instalación de librerías y dependencias

Después de haber iniciado la Jetson Nano, se abrió una terminal y se ingresaron los siguientes comandos para la instalación de las librerías a utilizar:

1. Actualizar el sistema

```
sudo apt-get update  
sudo apt-get upgrade
```

2. dlib

```
pip3 install dlib
```

3. boto3

```
sudo pip3 install boto3
```

4. imutils

```
pip3 install imutils
```

5. geocoder

```
pip3 install geocoder
```

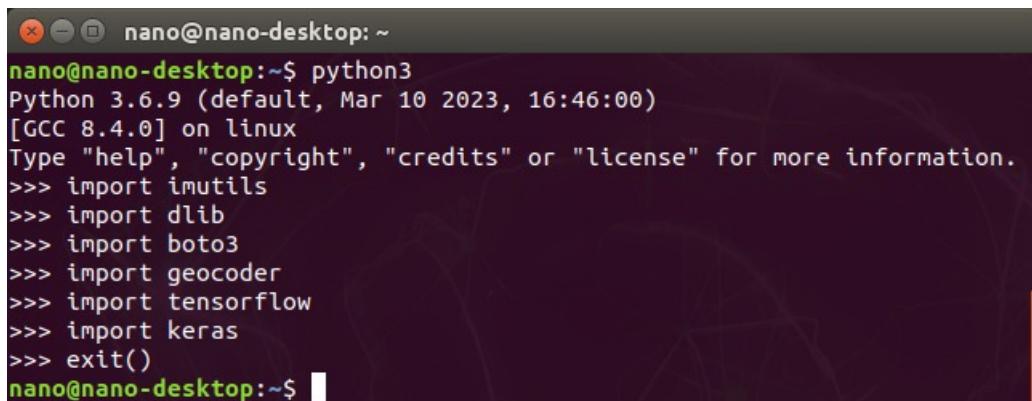
6. Tensorflow En el caso de Tensorflow se debe instalar una versión compatible con Python 3.6 y la versión instalada del JetPack [48].

```
sudo apt-get update
sudo apt-get install libhdf5-serial-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpeg8-dev
sudo apt-get install python3-pip
sudo pip3 install -U pip testresources setuptools
sudo ln -s /usr/include/locale.h /usr/include/xlocale.h
sudo pip3 install -U numpy==1.19.4 future mock keras_preprocessing keras_applications
sudo pip3 install --extra-index-url https://developer.download.nvidia.com/compute/redi
```

7. Keras

```
pip3 install keras==2.7.0
```

Se importaron las librerías con Python 3 para verificar que hayan sido correctamente instaladas.



The screenshot shows a terminal window titled "nano@nano-desktop: ~". The user runs the command "python3" which outputs the Python version (3.6.9) and build information. Then, the user imports several libraries one by one: imutils, dlib, boto3, geocoder, tensorflow, and keras. Finally, the user exits the session with "exit()".

```
nano@nano-desktop:~$ python3
Python 3.6.9 (default, Mar 10 2023, 16:46:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import imutils
>>> import dlib
>>> import boto3
>>> import geocoder
>>> import tensorflow
>>> import keras
>>> exit()
nano@nano-desktop:~$
```

Figura 52: Importación de librerías instaladas con Python.

Se instaló Jetson Stats, la cual es una herramienta de monitoreo del sistema que proporciona información de rendimiento de Jetson en tiempo real. Muestra información sobre el uso de la CPU, la GPU, la memoria y otros recursos del sistema. jtop muestra la salida de los monitores de energía Jetson y los sensores de temperatura [49].

Comandos usados para la instalación de Jetson Stats:

```
sudo apt install python3-pip
sudo pip3 install -U jetson-stats
sudo reboot
jtop
```

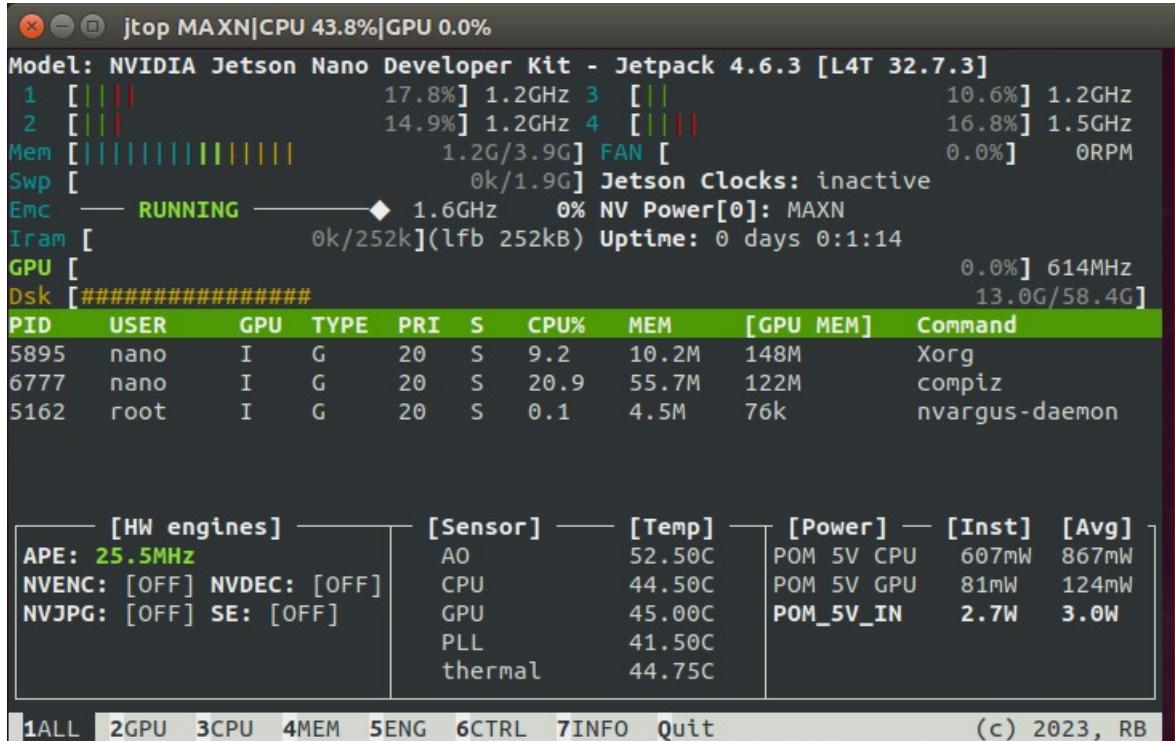


Figura 53: Interfaz despues de la instalación de Jetson Stats.

Archivo Swap

El swap es un espacio de intercambio, que bien puede ser una partición lógica en el disco o simplemente un archivo. En lugar de utilizar espacio en la memoria RAM, el swap utiliza espacio en el disco duro para almacenar datos temporales, reduciendo así el uso de la RAM. El conjunto combinado de la memoria RAM y el swap crea una memoria virtual mayor a la que trae el ordenador por defecto. Así, el Kernel de Linux puede ejecutar procesos que requieren más memoria de la que se encuentra físicamente disponible [50]. Comandos utilizados para configurar memoria virtual:

```
sudo systemctl disable nvzramconfig
sudo fallocate -l 4G /mnt/4G.swap
sudo mkswap /mnt/4G.swap
sudo swapon /mnt/4G.swap
```

```
cd ..
cd ..
cd etc/
```

```
sudo gedit fstab
```

El comando gedit abre un archivo que se debe modificar añadiendo la siguiente línea y posteriormente guardando los cambios realizados.

```
/mnt/4G.swap none swap sw 0 0
```

Luego, se verificó el tamaño de todos los intercambios:

```
free -h
```

Para habilitar la memoria zram de nuevo se utiliza el comando:

```
sudo systemctl enable nvzramconfig
```



	total	used	free	shared	buff/cache
available					
Mem:	3.9G	1.4G	833M	36M	1.7G
	2.3G				
Swap:	4.0G	0B	4.0G		

Figura 54: verificación del archivo swap credo.

OpenCV con CUDA (Arquitectura de dispositivo unificado de cómputo) Por defecto, la versión del JetPack SDK 4.6.3 viene instalada con OpenCV 4.1.1 sin CUDA, el cual es un marco que permite la computación paralela en la GPU. Para activar CUDA, se utilizó el script proporcionado en el repositorio de GitHub [51], el cual además instala la versión 4.5 de OpenCV:

```
git clone https://github.com/mdegans/nano_build_opencv.git
cd nano_build_opencv/
gedit build_opencv.sh
```

El archivo se modificó colocando la variable JOBS=4 después de haber modificado el archivo de intercambio (*swapfile*). Posteriormente, se ejecutó el archivo.

```
./build_opencv.sh 4.5.4
```



Figura 55: Visualización de la instalación exitosa de OpenCV con Cuda en Jtop.

8.1.2. Implementar algoritmo para la detección del rostro

En este fragmento de código, se lleva a cabo la inicialización de la cámara y se configura la instancia para la captura de video. Adicionalmente, se realiza la inicialización del detector de rostros, un componente esencial para el análisis de imágenes en el contexto de la aplicación. La configuración de la cámara se realiza mediante la cadena `gst_str` utilizando GStreamer, y posteriormente se establece la instancia de captura de video. Además, se emplea el detector de rostros proporcionado por la biblioteca Dlib, el cual desempeñará un papel fundamental en la detección y análisis de rostros en los fotogramas capturados por la cámara.

```
# Inicializar la cámara y tomar la instancia
gst_str = ('nvarguscamerasrc ! video/x-raw(memory:NVMM), width=(int)1280, height=(int)720, format=(string)NV12, framerate=(fraction)5/1 ! '
           'nvvidconv flip-method=0 ! video/x-raw, width=(int)640, height=(int)480, format=(string)BGR ! '
           'videoconvert ! video/x-raw, format=(string)BGR ! appsink')
cap = cv2.VideoCapture(gst_str, cv2.CAP_GSTREAMER)

# Inicialización del detector de rostros y el detector de puntos de referencia
#-----Modelos-----
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
```

Figura 56: Inicialización de cámara y detector.

`gst_str` contiene una cadena de configuración para la cámara utilizando GStreamer, una herramienta para construir flujos multimedia. Esta configuración está diseñada para trabajar con cámaras Nvidia Jetson, específicamente para el formato de memoria "NV12", el cual es un formato de píxeles comúnmente utilizado en dispositivos Nvidia y con una resolución de 1280x720 píxeles. Esto indica que cada fotograma capturado tendrá una dimensión de 1280 píxeles de ancho por 720 píxeles de alto.

```
while True:
    suc,frame = cap.read()

    if not suc :
        break

    #gps.get_gps_position()
    #-----FPS-----
    ctime = time.time()
    fps= int(1/(ctime-ptime))
    ptime = ctime
    cv2.putText(frame,f'FPS:{fps}',(frame.shape[1]-120,frame.shape[0]-20),cv2.FONT_HERSHEY_PLAIN,2,(0,200,0),3)

    #-----Deteccion de rostro-----
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = detector(gray)
    # Detectar cara en matriz de caras
    for face in faces:
        x1 = face.left()
        y1 = face.top()
        x2 = face.right()
        y2 = face.bottom()

        face_frame = frame.copy()
        cv2.rectangle(face_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
```

Figura 57: Implementación del detector de rostro.

El código anterior implementa un bucle infinito para capturar continuamente frames de video desde una cámara. Para cada frame capturado, realiza las siguientes acciones:

1. Calcula el tiempo actual y el número de fotogramas por segundo (FPS) con el objetivo de mostrar esta información en la esquina superior derecha del video.
2. Convierte el frame a escala de grises para facilitar la detección de rostros.
3. Utiliza un detector de rostros para identificar caras en el frame en escala de grises.
4. Itera sobre las caras detectadas y dibuja un rectángulo verde alrededor de cada cara en el frame original.

Este conjunto de operaciones se repite de forma continua, lo que resulta en un análisis en tiempo real de los frames de video para la detección y resaltado de rostros mediante rectángulos verdes en cada cara encontrada. Además, se muestra el FPS en el video para proporcionar información adicional sobre la velocidad de captura de los frames. El bucle se detiene si no se puede capturar un nuevo frame.

8.1.3. Implementar puntos de referencia en boca y ojos

Esta sección del código se encarga de detectar y marcar puntos de referencia faciales, conocidos como "landmarks", en las regiones de la boca y los ojos de cada cara detectada. Los landmarks se obtienen utilizando un predictor específico y se representan como coordenadas en un formato NumPy. Luego, se seleccionan los landmarks correspondientes a la boca, desde el índice 48 hasta el índice 59, lo cual abarca la región de la boca, y se dibuja un contorno alrededor de esta en el frame original. Además, se evalúa si los ojos están parpadeando mediante una función llamada `blinked`. Para el ojo izquierdo, se utilizan los landmarks desde el índice 36 hasta el índice 41, y para el ojo derecho, se emplean los landmarks desde el índice 42 hasta el índice 47 para calcular una métrica de parpadeo.

```
#-----Detectar Landmarks-----#
landmarks = predictor(gray, face)
landmarks = face_utils.shape_to_np(landmarks)

#-----Boca-----#
#-----Detectar/Marcar puntos de referencia del labio superior e inferior-----#
lip = landmarks[48:60]
cv2.drawContours(frame,[lip],-1,(0, 165, 255),thickness=3)

#-----OJOS-----#
# -----Detectar/Marcar puntos de referencia que mostrarán a los ojos-----#
left_blink = blinked(landmarks[36], landmarks[37],
                     landmarks[38], landmarks[41], landmarks[40], landmarks[39])
right_blink = blinked(landmarks[42], landmarks[43],
                      landmarks[44], landmarks[47], landmarks[46], landmarks[45])
```

Figura 58: Puntos de referencia facial.

8.1.4. Implementar metrica MOR y medición de parpadeos

Se crearon las siguientes funciones que tienen como objetivo realizar análisis relacionados con la detección de parpadeo y la medición de la apertura de la boca.

- Función `cal_yawn`: Esta función toma los *landmarks* faciales como entrada, identifica y extrae los *landmarks* que representan la parte superior e inferior de los labios, calcula el promedio de las posiciones de estos landmarks superiores e inferiores, y finalmente, calcula la distancia euclíadiana entre estos dos puntos promedio, proporcionando así una medida de la apertura de la boca.
- Función `compute`: Esta función calcula la distancia euclíadiana entre dos puntos dados, representados por las variables `ptA` y `ptB`.
- Función `blinked`: La función `blinked` toma como entrada seis *landmarks* correspondientes a los ojos, calcula las distancias entre puntos específicos para evaluar si el ojo está parpadeando, y retorna un valor numérico. Devuelve:
 - 2 si el parpadeo es significativo.
 - 1 si el parpadeo es moderado.
 - 0 si no se detecta un parpadeo significativo.

```
148 def cal_yawn(landmarks):
149     top_lip = landmarks[50:53]
150     top_lip = np.concatenate((top_lip, landmarks[61:64]))
151
152     low_lip = landmarks[56:59]
153     low_lip = np.concatenate((low_lip, landmarks[65:68]))
154
155     top_mean = np.mean(top_lip, axis=0)
156     low_mean = np.mean(low_lip, axis=0)
157
158     distance = dist.euclidean(top_mean,low_mean)
159     return distance
160
161
162 def compute(ptA, ptB):
163     dist = np.linalg.norm(ptA - ptB)
164     return dist
165
166
167 def blinked(a, b, c, d, e, f):
168     up = compute(b, d) + compute(c, e)
169     down = compute(a, f)
170     ratio = up/(2.0*down)
171
172     # Comprobando si parpadea
173     if (ratio > 0.25):
174         return 2
175     elif (ratio > 0.21 and ratio <= 0.25):
176         return 1
177     else:
178         return 0
```

Figura 59: Implementación de la metrica MOR y medición de parpadeos.

8.1.5. Implementar algoritmo para la detección somnolencia

Esta sección implementa el código para realizar el seguimiento y análisis en tiempo real de un flujo de video capturado por una cámara para clasificar un estado de somnolencia.

1. Cálculo de la Apertura de la Boca:

- Utiliza la función `cal_yawn` para calcular la distancia entre los landmarks asociados a la parte superior e inferior de los labios, proporcionando una medida de la apertura de la boca.
- Compara esta medida con un umbral (`yawn_thresh`) para determinar si se ha superado, indicando la posibilidad de un bostezo.
- Registra el inicio y finalización de un bostezo, calculando su duración.

```
lip_dist = cal_yawn(landmarks)
#print(lip_dist)

if lip_dist > yawn_thresh :
    print("Limite superado, valor: " + str(lip_dist))
    if not is_yawning:
        is_yawning = True
        start_time_yawn = time.time() # Tiempo de inicio del bostezo
    else:
        if is_yawning:
            is_yawning = False
            end_time_yawn = time.time() # Tiempo de finalización del bostezo
            yawn_duration = end_time_yawn - start_time_yawn
            print("Duración del bostezo:", yawn_duration, "segundos")
```

Figura 60: Cálculo de la Apertura de la Boca.

Detección de Parpadeo:

- Utiliza la función `blinked` para evaluar si los ojos están parpadeando.
- Registra el número de parpadeos significativos y realiza acciones específicas según el estado de alerta del sujeto (dormido, somnoliento o activo).

```

if (left_blink == 0 or right_blink == 0):
    sleep += 1
drowsy = 0
active = 0
if (sleep > 4):
    status = "Dormido"
    color = (255,0,0)
    if start_recording == 0:
        start_recording = time.time()
        video = cv2.VideoWriter(file, fourcc, 4, (640, 480))
        print("Grabando video...")
        video.write(frame)
    if time.time() - start_recording > 5:
        start_recording = 0
        video.release()
        upload_video[file]
elif (left_blink == 1 or right_blink == 1):
    sleep = 0
    active = 0
    drowsy += 1
    if (drowsy > 4):
        status = "Somnoliento"
        color = (0, 0, 255)
        color = (255,0,0)
        if start_recording == 0:
            start_recording = time.time()
            video = cv2.VideoWriter(file, fourcc, 4, (640, 480))
            print("Grabando video...")
            video.write(frame)
        if time.time() - start_recording > 5:
            start_recording = 0
            video.release()

```

Figura 61: Detección de Parpadeo.

Alertas y Grabación de Video:

- Genera alertas en casos específicos, como detectar más de dos bostezos por minuto.
- Inicia la grabación de video en situaciones de interés, como cuando somnolencia.

```

#-----Deteccion de bostezos por minuto-----
if elapsed_time >= 60: # Si ha pasado 1 minuto
    yawn_count = 0 # Reiniciar el contador de bostezos
    start_time = current_time # Reiniciar el tiempo de inicio

if elapsed_time < 60: # Si se detectan mas de 2 bostezos envia alerta
    if yawn_count >= 2:
        print("¡ALERTA! Se han detectado más de 2 bostezos por minuto.")

        #grabar video
        #video.write(frame)

        #upload_video(file)

#-----Conteo de Bostezos-----
if yawn_duration >= 5: # Si la duración del bostezo es igual o mayor a 5 segundos se considera bostezo
    yawn_count += 1
    status = "Bostezo !!!"
    color = (255, 255, 0)
    yawn_duration = 0 # Reiniciar la duración del bostezo

# iterar sobre los 68 landmarks detectados en el rostro y dibujar un círculo en cada uno de ellos.
for n in range(0, 68):
    (x, y) = landmarks[n]
    cv2.circle(face_frame, (x, y), 1, (255, 255, 255), -1)

```

Figura 62: Detección de Parpadeo.

Visualización en Tiempo Real:

- Dibuja un círculo en cada landmark facial detectado para visualizar los puntos de referencia en el rostro.
- Muestra el estado actual (dormido, somnoliento, activo) y la duración de los bostezos en el video.

```
# iterar sobre los 68 landmarks detectados en el rostro y dibujar un círculo en cada uno de ellos.
for n in range(0, 68):
    (x, y) = landmarks[n]
    cv2.circle(face_frame, (x, y), 1, (255, 255, 255), -1)

cv2.imshow('CSI camera', frame) # mostrar imagen en una ventana.
if cv2.waitKey(1) & 0xFF == ord('q') : # presionar la tecla 'q' para deter la ejecución del programa.
    break
if video is not None:
    video.release()
```

Figura 63: Visualización en Tiempo Real.

Conteo de Bostezos y FPS:

- Realiza un seguimiento del conteo de bostezos por minuto.
- Muestra los fotogramas por segundo (FPS) en la esquina superior derecha del video.

```
#gps.power_on(power_key)
while True:
    suc,frame = cap.read()

    if not suc :
        break

    #gps.get_gps_position()
    #-----FPS-----
    ctime = time.time()
    fps= int(1/(ctime-ptime))
    ptime = ctime
    cv2.putText(frame,f'FPS:{fps}',(frame.shape[1]-120,frame.shape[0]-20),cv2.FONT_HERSHEY_PLAIN,2,(0,200,0),3)
```

Figura 64: Visualización de FPS.

8.1.6. Integración de los periféricos y accesorios al Módulo Central de Procesamiento

Se integraron los periféricos dentro del contenedor para la Jetson Nano.

El contenedor de la Jetson Nano es una carcasa de metal con un ventilador de refrigeración de 5 V, diseñado específicamente para ser compatible con la placa de desarrollo NVIDIA Jetson Nano (versión B01 con 4 GB de RAM).

Esta carcasa ofrece varias características adicionales, como soporte para reinicio de cámara, un botón de encendido y compatibilidad específica con la cámara Waveshare IMX219 Series. Además, el contenedor también tiene soporte para adaptadores inalámbricos Wireless-AC8265, lo que permite la conectividad Wi-Fi y Bluetooth en la Jetson Nano. Otra característica destacada es la compatibilidad con el módulo de placa de expansión LTE CAT4 basado en SIM7600G-H. Este módulo proporciona capacidades de comunicación inalámbrica, como llamadas telefónicas, mensajes de texto (SMS) y posicionamiento GPS utilizando los sistemas BeiDou y Glonass.



Figura 65: Carcasa sin armar.

Contenedor de la Jetson Nano junto con los periféricos integrados:



Figura 66: Carcasa armada con la Jetson Nano y periféricos integrados.

8.2. Módulo de Comunicaciones

8.2.1. Investigación de la documentación del módulo 3G/4G LTE-Base Hat

Para el presente proyecto, se hará uso del Base-Hat SIM7600G-H 4G para Jetson Nano.



Figura 67: Módulo Base-Hat SIM7600G-H

En primer lugar, utilizando la terminal del sistema operativo Ubuntu, se ingresan los siguientes comandos:

```
sudo apt-get update
sudo apt-get install python3-pip
sudo pip3 install pyserial
mkdir -p ~/Documents/SIM7600X_4G_for_JETSON_NANO
wget -P ~/Documents/SIM7600X_4G_for_JETSON_NANO/ https://www.waveshare.com/w/upload/6/64/SIM7600X_
4G_for_JETSON_NANO.tar.gz
cd ~/Documents/SIM7600X_4G_for_JETSON_NANO/
tar -xvf SIM7600X_4G_for_JETSON_NANO.tar.gz
sudo pip3 install Jetson.GPIO
sudo groupadd -f -r gpio
sudo usermod -a -G gpio your_user_name
sudo udevadm control --reload-rules && sudo udevadm trigger
sudo apt-get install minicom
```

Figura 68: Instalación de librerías

Los comandos ingresados en la figura 149 se encargan de instalar todas las bibliotecas y el software necesario para poder comenzar a utilizar la red LTE mediante el módulo SIM7600G-H. Además, también se crea un directorio que contendrá la configuración de usuario, así como una cuenta enlazada al módulo.

Posteriormente, probamos que el puerto GPIO de nuestra Jetson Nano esté funcionando con los siguientes comandos:

```
echo 200 > /sys/class/gpio/export
echo out > /sys/class/gpio200/direction
echo 1 > /sys/class/gpio200/value
echo 0 > /sys/class/gpio200/value
```

Figura 69: Instalación de librerías

Después de haber realizado los pasos anteriores, el pin con el nombre NET deberá parpadear constantemente, lo que significa que el módulo está listo para ser utilizado.

Para realizar la comunicación LTE, primero se ingresa a la librería minicom utilizando los siguientes comandos:

```
sudo su
killall ModemManager
minicom -D /dev/ttyUSB2
```

Figura 70: Minicom

Posteriormente, se necesita actualizar los drivers:

```
-----  
cd  
wget https://www.waveshare.com/w/upload/4/46/Simcom_wwan.zip  
unzip Simcom_wwan.zip  
cd simcom_wwan  
sudo su  
make
```

Figura 71: Actualización de drivers

Finalmente se establece una dirección IP con el siguiente comando:

- Allocate IP

```
apt-get install udhcpc  
udhcpc -i wwan0
```

Figura 72: Establecer dirección IP

8.2.2. Implementación del sistema de Geolocalización

Habilitar GPS y obtener datos de posición GPS desde la Jetson Nano [52]

Se inicia abriendo una terminal e instalando bibliotecas con los siguientes comandos.

Se remplaza `your_user_name` con el nombre de usuario nano.

```
sudo apt-get update  
sudo apt-get install python3-pip  
sudo pip3 install pyserial  
mkdir -p ~/Documents/SIM7600X_4G_for_JETSON_NANO  
wget -P ~/Documents/SIM7600X_4G_for_JETSON_NANO/ https://www.waveshare.c  
om/w/upload/6/64/SIM7600X_4G_for_JETSON_NANO.tar.gz  
cd ~/Documents/SIM7600X_4G_for_JETSON_NANO/  
tar -xvf SIM7600X_4G_for_JETSON_NANO.tar.gz  
sudo pip3 install Jetson.GPIO  
sudo groupadd -f -r gpio  
sudo usermod -a -G gpio your_user_name  
sudo udevadm control --reload-rules && sudo udevadm trigger  
sudo apt-get install minicom
```

Figura 73: Instalación de bibliotecas

```
echo 200 > /sys/class/gpio/export
echo out > /sys/class/gpio200/direction
echo 1 > /sys/class/gpio200/value
echo 0 > /sys/class/gpio200/value
```

Figura 74: Instalaci?n de comandos

Se coloca el DIP en ON y se ejecuta minicom con el comando AT para probar el SIM7600.

```
sudo minicom -D /dev/ttyTHS1 -b 115200
```

Figura 75: Prueba de SIM7600G

Ejemplo de GPS: Se conectó la antena GPS y se colocó el receptor en un lugar abierto. Se ejecutaron los siguientes comandos para probar el GPS.

```
cd ~/Documents/SIM7600X_4G_for_JETSON_NANO/GPS/
sudo python3 GPS.py
```

Figura 76: Comandos GPS para pruebas.

Comunicación por Minicom:

Para obtener los datos de posición GPS, fue necesario interactuar con el módulo SIM7600X a través del programa en Python:

```
GPS.py > getGpsPosition
1  #!/usr/bin/python
2
3  import Jetson.GPIO as GPIO
4  import serial
5  import time
6
7  ser = serial.Serial('/dev/ttyTHS1',115200)
8  ser.flushInput()
9
10 powerKey = 6
11 rec_buff = ''
12 rec_buff2 = ''
13 time_count = 0
14
15 def sendAt(command,back,timeout):
16     #rec_buff = ''
17     ser.write((command+'\r\n').encode())
18     time.sleep(timeout)
19     if ser.inWaiting():
20         time.sleep(0.01 )
21         rec_buff = ser.read(ser.inWaiting())
22     if rec_buff != '':
23         if back not in rec_buff.decode():
24             print(command + ' ERROR')
25             print(command + ' back:\t' + rec_buff.decode())
26             return 0
27         else:
28             print(rec_buff.decode())
29             return 1
30     else:
31         print('GPS is not ready')
32         return 0
```

Figura 77: C?digo que permite obtener la posici?n GPS parte 1

```

GPS.py > ...
33
34     def getGpsPosition():
35         rec_null = True
36         answer = 0
37         print('Start GPS session...')
38         rec_buff = ''
39         sendAt('AT+CGPS=1,1','OK',1)
40         time.sleep(2)
41         while rec_null:
42             answer = sendAt('AT+CGPSINFO','+CGPSINFO: ',1)
43             if 1 == answer:
44                 answer = 0
45                 if ',,,,,,,' in rec_buff:
46                     print('GPS is not ready,wait 10 seconds')
47                     rec_null = False
48                     time.sleep(10)
49             else:
50                 print('error %d'%answer)
51                 rec_buff = ''
52                 sendAt('AT+CGPS=0','OK',1)
53                 return False
54         time.sleep(1.5)
55         #整理数据内容
56
57     def powerOn(powerKey):
58         print('SIM7600X is starting:')
59         GPIO.setmode(GPIO.BCM)
60         GPIO.setwarnings(False)
61         GPIO.setup(powerKey,GPIO.OUT)
62         time.sleep(0.1)
63         GPIO.output(powerKey,GPIO.HIGH)

```

Figura 78: C?digo que permite obtener la posici?n GPS parte 2

```

GPS.py > ⚡ getGpsPosition
63     time.sleep(0.1)
64     GPIO.output(powerKey,GPIO.HIGH)
65     time.sleep(2)
66     GPIO.output(powerKey,GPIO.LOW)
67     time.sleep(20)
68     ser.flushInput()
69     print('SIM7600X is ready')
70
71     def powerDown(powerKey):
72         GPIO.setmode(GPIO.BCM)
73         GPIO.setwarnings(False)
74         GPIO.setup(powerKey,GPIO.OUT)
75         print('SIM7600X is loging off:')
76         GPIO.output(powerKey,GPIO.HIGH)
77         time.sleep(3)
78         GPIO.output(powerKey,GPIO.LOW)
79         time.sleep(18)
80         print('Good bye')
81
82     def checkStart():
83         while True:
84             ser.write( ('AT\r\n').encode() )
85             time.sleep(0.1);
86             if ser.inWaiting():
87                 time.sleep(0.01)
88                 recBuff = ser.read(ser.inWaiting())
89                 print( 'try to start\r\n' + recBuff.decode() )
90                 if 'OK' in recBuff.decode():
91                     recBuff = ''
92                     return
93             else:

```

Figura 79: C?digo que permite oibtener la posici?n GPS parte 3

```
93     else:
94         powerOn(powerKey)
95         time.sleep(1)
96     try:
97         checkStart()
98         getGpsPosition()
99         powerDown(powerKey)
100    except:
101        if ser != None:
102            ser.close()
103            powerDown(powerKey)
104            GPIO.cleanup()
105    if ser != None:
106        ser.close()
107        GPIO.cleanup()
108
```

Figura 80: Código que permite obtener la posición GPS parte 4

AWS Maps

AWS Maps es un servicio de Amazon Web Services que proporciona capacidades de mapas y geolocalización en la nube. Permite crear, visualizar y analizar datos geoespaciales en aplicaciones y servicios.

Se creó un mapa en AWS Maps donde se configuró la apariencia y funcionalidad del mismo, incluyendo los estilos de visualización, las capas de datos y las interacciones del usuario. El propósito de este mapa es mostrar información geográfica, como ubicaciones, rutas, geovallas, puntos de interés, entre otros. La configuración del mapa permitió personalizar su aspecto y adaptarlo a las necesidades del proyecto.

Para crear el mapa, se inició sesión en la Consola de administración de AWS. Luego, se accedió a la página de AWS Maps Service en la consola de administración. A continuación, se hizo clic en “Crear mapa” para comenzar a crearlo. En el proceso de creación, se seleccionó un nombre para el mapa y se especificó la región de AWS donde se deseaba crearlo. Seguidamente, se eligió el estilo de mapa y posteriormente se configuraron las capas del mapa, pudiendo agregar capas para mostrar datos como carreteras, edificios, ríos, y también se pudieron agregar datos geoespaciales personalizados como capas personalizadas. Finalmente, se creó el mapa.

Una vez creado el mapa en AWS Maps, se agregó y configuró un rastreador para realizar el seguimiento del dispositivo llamado ”mydevice.” en el mapa. Los rastreadores se encargaron de recopilar y actualizar la ubicación de los dispositivos, brindando información en tiempo real.

A continuación, se presenta un resumen de cómo se creó un rastreador en AWS Maps:

En la página del mapa en la consola de administración de AWS Maps, se hizo clic en “Crear rastreador”. Se seleccionó un nombre para el rastreador y se especificó la región de AWS donde se deseaba crearlo. Se procedió a configurar los detalles del rastreador, como la frecuencia de actualización de la ubicación de los dispositivos, en este caso, cada 10 minutos se envían las coordenadas. Finalmente, se hizo clic en “Crear rastreador” para finalizar la configuración del mismo.

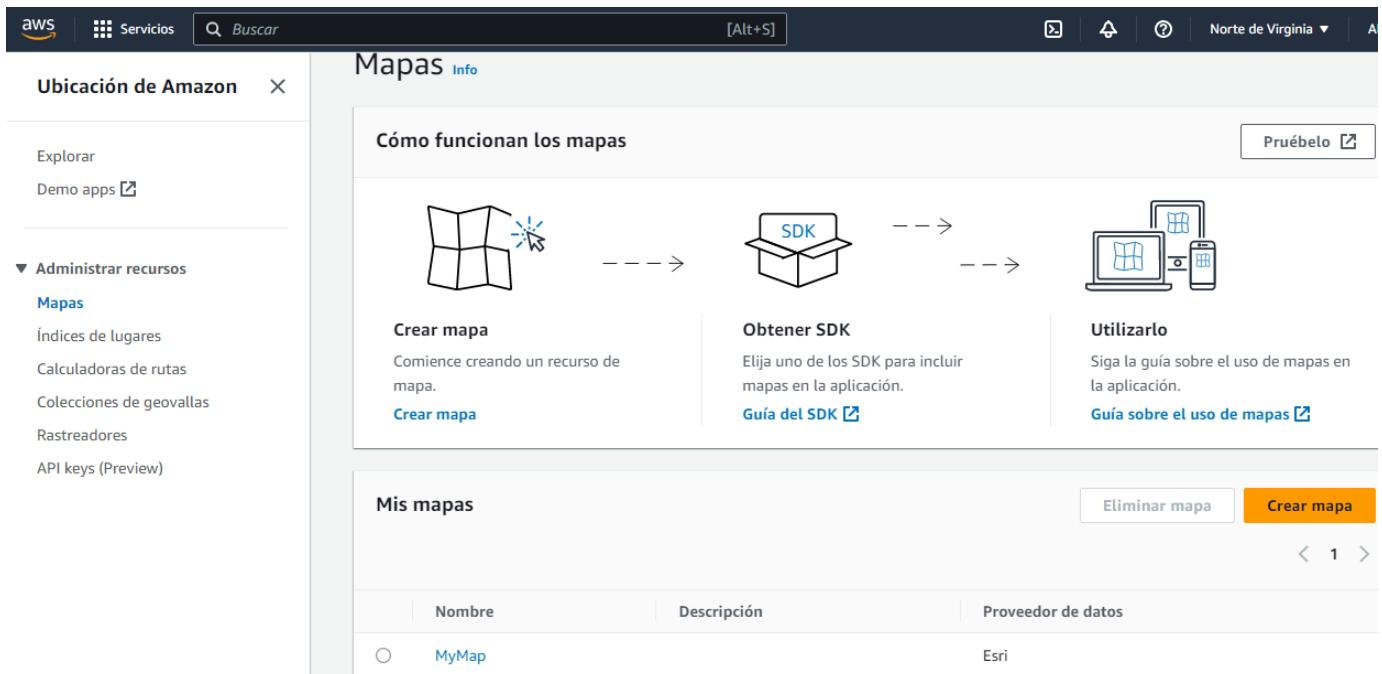


Figura 81: Mapa creado en AWS - MyMap

Ubicación geográfica en la Aplicación web

Para obtener el historial de posiciones de un dispositivo, se definió la función llamada "getDevicePosition". Posteriormente, se utilizó la función `client.getPositionHistory` con el argumento "params" para solicitar el historial de posiciones del dispositivo. Después, se creó la variable "tempPosMarkers" para almacenar los resultados obtenidos al mapear los elementos de "data.DevicePositions". Para cada elemento "devPos" en "data.DevicePositions", se imprimió su posición en la consola y se generó un nuevo objeto con propiedades "index", "Long" y "lat", que representaban los valores de la posición del dispositivo. Este archivo se guardó con la extensión .jsx.



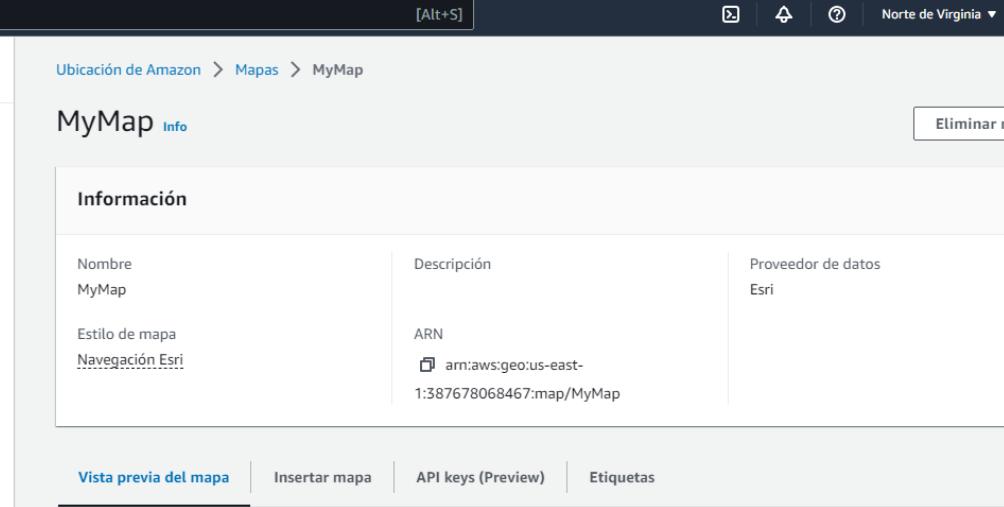
```

1  const getDevicePosition = () => {
2      setDevPosMarkers([]);
3
4      var params = {
5          DeviceId: deviceID,
6          TrackerName: trackerName,
7          StartTimeInclusive: "2023-05-12T10:05:07.327Z" ,
8          EndTimeExclusive: new Date()
9      };
10
11     client.getDevicePositionHistory(params, (err, data) => {
12         if (err) console.log(err, err.stack);
13         if (data) {
14             console.log(data);
15             const tempPosMarkers = data.DevicePositions.map( function (devPos, index) {
16                 console.log(devPos.Position[0], devPos.Position[1])
17                 return {
18                     index: index,
19                     long: devPos.Position[0],
20                     lat: devPos.Position[1]
21                 }
22             });
23         });

```

Figura 82: Código para obtener la ubicación geográfica en la aplicación web

Obtención de coordenadas geográficas desde la Jetson Nano



The screenshot shows the AWS Maps interface. On the left, there's a sidebar with navigation links like 'Ubicación de Amazon', 'Explorar', 'Demo apps', and a 'Mapas' section under 'Administrar recursos'. The main area displays a map titled 'MyMap' with a coordinate marker. Below the map, there's an 'Información' card with the following details:

Nombre	Descripción	Proveedor de datos
MyMap		Esri
Estilo de mapa Navegación Esri	ARN arn:aws:geo:us-east-1:387678068467:map/MyMap	

At the bottom, there are tabs for 'Vista previa del mapa', 'Insertar mapa', 'API keys (Preview)', and 'Etiquetas'. The 'Vista previa del mapa' tab is currently selected.

Figura 83: Coordenadas geográficas en la Jetson Nano

Mapa creado en AWS Maps

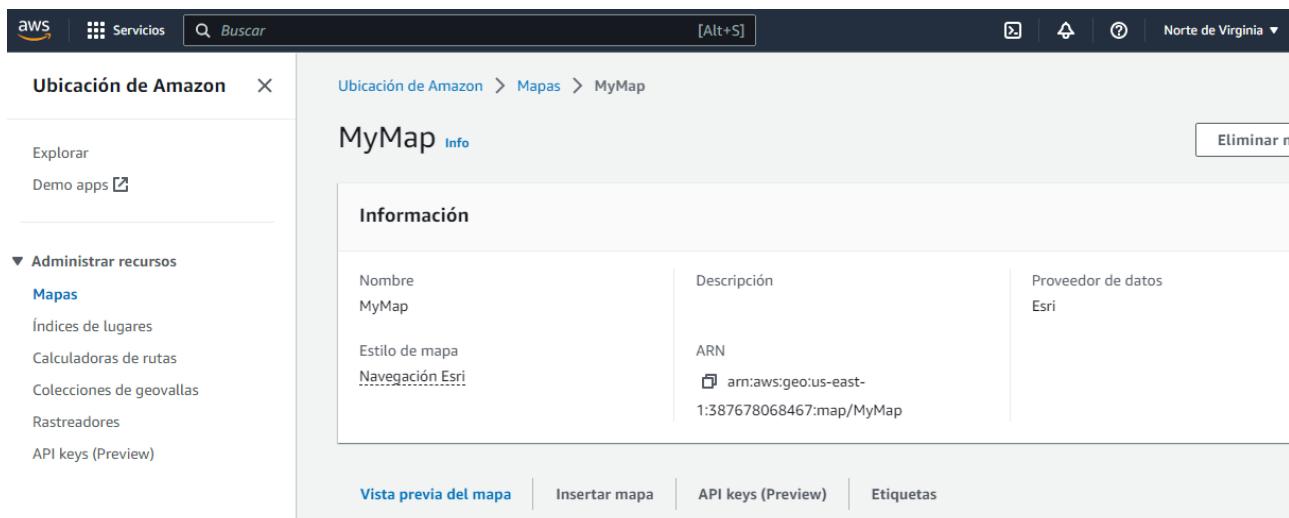


Figura 84: Detalles del mapa creado en AWS

coordenadas geográficas que recibe la página web

```

▶ {DevicePositions: Array(1)}
-99.14154126666666 19.4703304
▶ {DevicePositions: Array(1)}
-99.14154126666666 19.4703304
▶ |
```

Figura 85: Coordenadas geográficas que recibe la página web

8.2.3. Conexión de la aplicación web con el Módulo Central de Procesamiento

La primer configuración que se realiza, fue la habilitación y activación de la red 4G Lte en la Jetson Nano por medio de la interfaz wwan0 [52] , que nos permite conectarnos a internet por medio de la red celular, se utiliza una SIM Telcel para adquirir GB y poder acceder a internet.

Se abre una terminal y se ejecutaron los siguientes comandos para abrir minicom por comando:

```

sudo su
killall ModemManager
minicom -D /dev/ttyUSB2
```

Figura 86: Abrir minicom por comando.

El siguiente comando se utiliza para verificar la conexión.

```
AT+CNMP=38
AT+CSQ
AT+CREG?
AT+COPS?
AT+CPSI?
```

Figura 87: Verificar conexión.

Posteriormente se descarga el controlador con los siguientes comandos:

```
cd
wget https://www.waveshare.com/w/upload/4/46/Simcom_wwan.zip
unzip Simcom_wwan.zip
cd simcom_wwan
sudo su
make
```

Figura 88: Descarga del driver.

Se usa el permiso de root para instalar el controlador:

```
insmod simcom_wwan.ko
lsmod
dmesg
```

Figura 89: Instalar driver.

Se comprueba si se reconoce la interfaz wwan0.

```
ifconfig -a
```

Figura 90: Comando para comprobar interfaz wwan0.

Se habilita la interfaz wwan0.

```
ifconfig wwan0 up
```

Figura 91: Habilitar la interfaz wwan0.

Se realiza la comunicación por Minicom:

```
minicom -D /dev/ttyUSB2  
AT$QCRMCALL=1,1
```

Figura 92: Comunicar por Minicom.

Se asigna la IP:

```
apt-get install udhcpc  
udhcpc -i wwan0
```

Figura 93: Asignar IP.

Posteriormente se realiza la conexión del Módulo Central de Procesamiento y la aplicaciónn web, se utiliza la librería **Boto3**

Boto3 es una biblioteca de Python desarrollada por Amazon Web Services (AWS) que permite interactuar con los servicios de AWS. Proporciona una interfaz de programación de aplicaciones (API) fácil de usar para acceder y administrar recursos en la nube de AWS [?].

Para crear una conexión a AWS Amplify utilizando Boto3, se siguieron los siguientes pasos :

En primer lugar, se instaló Boto3 utilizando el comando: *pip install boto3*. A continuación, se configuraron las credenciales de AWS, proporcionando las claves de acceso de IAM que otorgan acceso a los servicios de AWS. Estas credenciales se configuraron manualmente en el archivo de configuración. Luego, se importa el módulo Boto3 en el script de Python y se crea una conexión al servicio de AWS Amplify utilizando las credenciales configuradas anteriormente. A partir de la conexión establecida, se pudieron utilizar los métodos proporcionados por el cliente de Boto3 para interactuar con AWS Amplify.

ed 4G LTE activa en la Jetson Nano

```

nano@nano-desktop:~$ sudo systemctl enable simcom_wwan@wwan0.service
[sudo] password for nano:
nano@nano-desktop:~$ sudo systemctl start simcom_wwan@wwan0.service
nano@nano-desktop:~$ sudo systemctl status simcom_wwan@wwan0.service
● simcom_wwan@wwan0.service - 4G LTE network interface wwan0
   Loaded: loaded (/lib/systemd/system/simcom_wwan@.service; indirect; vendor pr
   Active: active (exited) since Wed 2023-05-17 16:52:10 CST; 5min ago
     Main PID: 7285 (code=exited, status=0/SUCCESS)
       Tasks: 1 (limit: 4181)
      CGroup: /system.slice/system-simcom_wwan.slice/simcom_wwan@wwan0.service
              └─7385 /sbin/dhclient -1 wwan0

may 17 16:51:55 nano-desktop systemd[1]: Starting 4G LTE network interface wwan0
may 17 16:52:10 nano-desktop bash[4755]: [+] wwan0 at /dev/ttyUSB2 is ready for
may 17 16:52:10 nano-desktop dhclient[7285]: DHCPDISCOVER on wwan0 to 255.255.25
may 17 16:52:10 nano-desktop dhclient[7285]: DHCPREQUEST of 10.30.139.38 on wwan
may 17 16:52:10 nano-desktop dhclient[7285]: DHCPOFFER of 10.30.139.38 from 10.3
may 17 16:52:10 nano-desktop dhclient[7285]: DHCPACK of 10.30.139.38 from 10.30.
may 17 16:52:10 nano-desktop dhclient[7285]: bound to 10.30.139.38 -- renewal in
may 17 16:52:10 nano-desktop systemd[1]: Started 4G LTE network interface wwan0.

Now you can use 4G network!

```

Installing as systemd service

Figura 94: Red 4G LTE activa por medio de la interfaz wwan0

Conexión a AWS Amplify utilizando Boto3

```

#credenciales

dynamodb = boto3.resource('dynamodb', aws_access_key_id='AKIAVUQ3J33Z7NM07M55',
                           aws_secret_access_key='[REDACTED]',
                           region_name='us-east-1')
table = dynamodb.Table('Incidencia-trjjwxbd3bbjrjaappwd23ijpi-dev')

location = geocoder.ip('me')
print(location.latlng)
coordenadas = json.loads(json.dumps(location.latlng), parse_float=Decimal)

nombre = 'prueba'
file = nombre + '.avi'
id = str(uuid.uuid1())
url_video = 'https://d3gh7t05x84ron.cloudfront.net/' + id + '.mp4'
cam = cv2.VideoCapture(0)
video = VideoWriter(file, VideoWriter_fourcc(*'XVID'), 15, (640,480))

```

Figura 95: Conexión con boto3 hacía AWS

8.3. Módulo de Estación Base

8.3.1. Definición de rutas del frontend

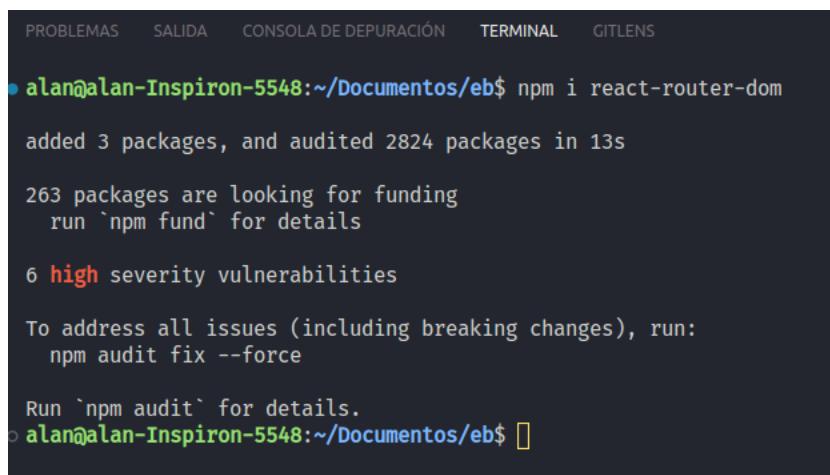
Rutas públicas vs rutas protegidas

Cuando se habla de una ruta protegida en React, se referiere a programar un bloqueo en ciertas rutas a la cual se le restringe el acceso al usuario. Esto comunmente se realiza para la validación de inicio de sesión de usuarios. Si el usuario no tiene una sesión iniciada, no podrá acceder a las rutas protegidas de la aplicación. Por otro lado, las rutas públicas son todas aquellas las cuales no requieren contar con una sesión iniciada, y pueden ser accesadas por cualquier tipo de usuario[?]. Como primer paso, se creo un proyecto de React utilizando el siguiente comando:

```
• alan@alan-Inspiron-5548:~/Documentos/eb$ npx create-react-app eb
```

Figura 96: Creación proyecto de react

Posteriormente, se realizó la instalación del modelo *React Router Dom* utilizando el siguiente comando:



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS

• alan@alan-Inspiron-5548:~/Documentos/eb$ npm i react-router-dom
added 3 packages, and audited 2824 packages in 13s

263 packages are looking for funding
  run `npm fund` for details

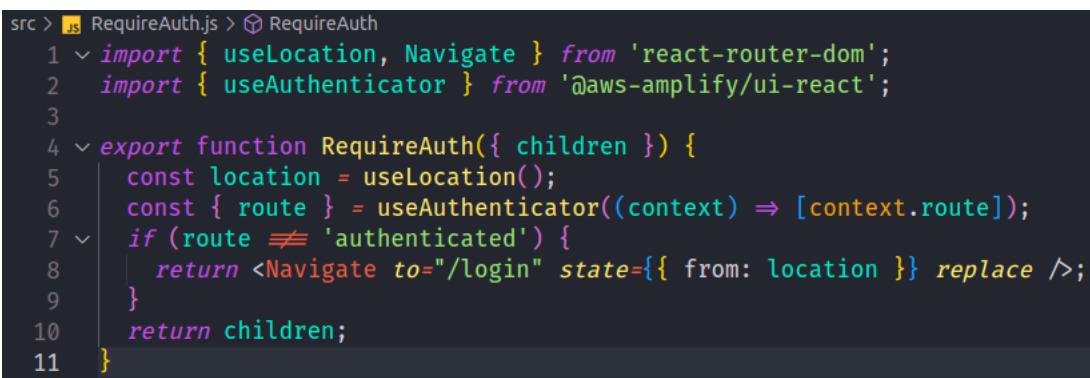
6 high severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
• alan@alan-Inspiron-5548:~/Documentos/eb$ 
```

Figura 97: Instalación React Router DOM

Utilizando la librería *useAuthenticator* ofrecida por el paquete de React Dom se creó un archivo de nombre *RequireAuth.js* el cuál contiene una función la cual se encargará de validar si existe una sesión iniciada previamente.



```
src > ↗ RequireAuth.js > ⚡ RequireAuth
  1  import { useLocation, Navigate } from 'react-router-dom';
  2  import { useAuthenticator } from '@aws-amplify/ui-react';
  3
  4  export function RequireAuth({ children }) {
  5    const location = useLocation();
  6    const { route } = useAuthenticator(({ context }) => [context.route]);
  7    if (route !== 'authenticated') {
  8      return <Navigate to="/login" state={{ from: location }} replace />;
  9    }
 10   return children;
 11 }
```

Figura 98: Función RequireAuth

Posteriormente, se necesita contar con una página de Login, la cuál permite validar que se encuentre una sesión iniciada por parte del usuario, para así poder acceder a las rutas protegidas.

```
src > Login.js > ...
1 import { useEffect } from "react";
2 import { Authenticator, useAuthenticator, View } from '@aws-amplify/ui-react';
3 import '@aws-amplify/ui-react/styles.css';
4 import { useNavigate, useLocation } from 'react-router';
5
6 export function Login() {
7   const { route } = useAuthenticator((context) => [context.route]);
8   const location = useLocation();
9   const navigate = useNavigate();
10  let from = location.state?.from?.pathname || '/';
11  useEffect(() => {
12    if (route === 'authenticated') {
13      navigate(from, { replace: true });
14    }
15  }, [route, navigate, from]);
16  return (
17    <View className="auth-wrapper">
18      <Authenticator></Authenticator>
19    </View>
20  );
21}
```

Figura 99: Componente Login

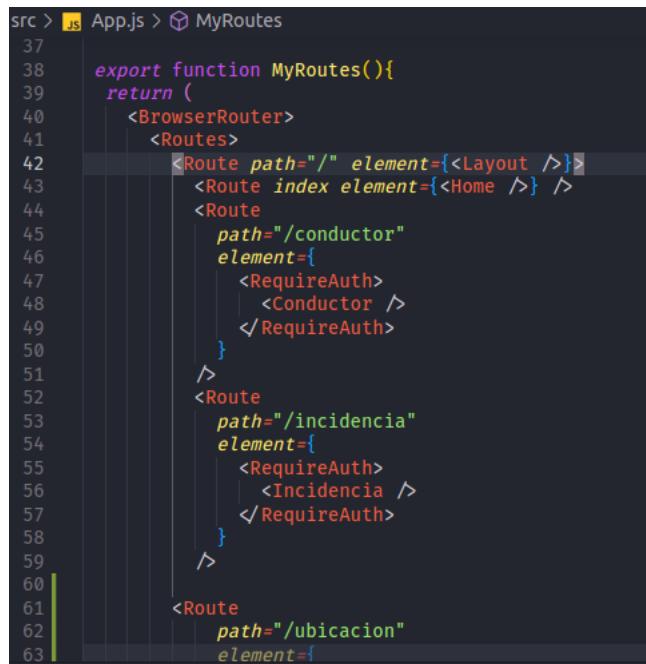
Para indicar a React, que se desea implementar una ruta protegida, se necesita ir al componente de dicha ruta e ingresar el siguiente código:

```
src > Home.js > Home
1 import { useAuthenticator, Authenticator } from "@aws-amplify/ui-react";
2
3 export default function Home() {
4   const { route } = useAuthenticator((context) => [context.route]);
5   const message =
6     route === 'authenticated' ? 'FIRST PROTECTED ROUTE!' : 'Loading ... ';
7   return (
8     <Authenticator>
9       {({ signOut, user }) => (
10         <main>
11           <h1>Bienvenido a Home</h1>
12
13           <button onClick={signOut}>Sign out</button>
14         </main>
15       )}
16     </Authenticator>
17   );
18
19
20}
```

Figura 100: Ruta protegida del componente Home

En dicho componente, se hace uso de las librerías `useAuthenticator` y `Authenticator` las cuales son ofrecidas por los servicios de Amazon Amplify. Se realizó esto para todos los componentes que se requirieron mantener como rutas protegidas.

Finalmente, dentro de nuestro componente `App.js`, se creó una función que contiene el directorio de rutas tanto públicas como protegidas:



```
src > js App.js > MyRoutes
37
38     export function MyRoutes(){
39         return (
40             <BrowserRouter>
41                 <Routes>
42                     <Route path="/" element={<Layout />}>
43                         <Route index element={<Home />} />
44                         <Route
45                             path="/conductor"
46                             element={
47                                 <RequireAuth>
48                                     <Conductor />
49                                 </RequireAuth>
50                             }
51                         />
52                         <Route
53                             path="/incidencia"
54                             element={
55                                 <RequireAuth>
56                                     <Incidencia />
57                                 </RequireAuth>
58                             }
59                         />
60                         <Route
61                             path="/ubicacion"
62                             element={
```

Figura 101: Directorio de rutas

Las rutas protegidas, están dentro de las etiquetas `<RequireAuth>`/`</RequireAuth>`, mientras que las públicas, se encuentran dentro de las etiquetas `<Route>`/`</Route>`.

Como resultado de todo lo anterior, se obtuvo una página de Login que utilizando los servicios de AWS Amplify y Cognito, permitirá iniciar sesión así como registrar a nuevos usuarios.

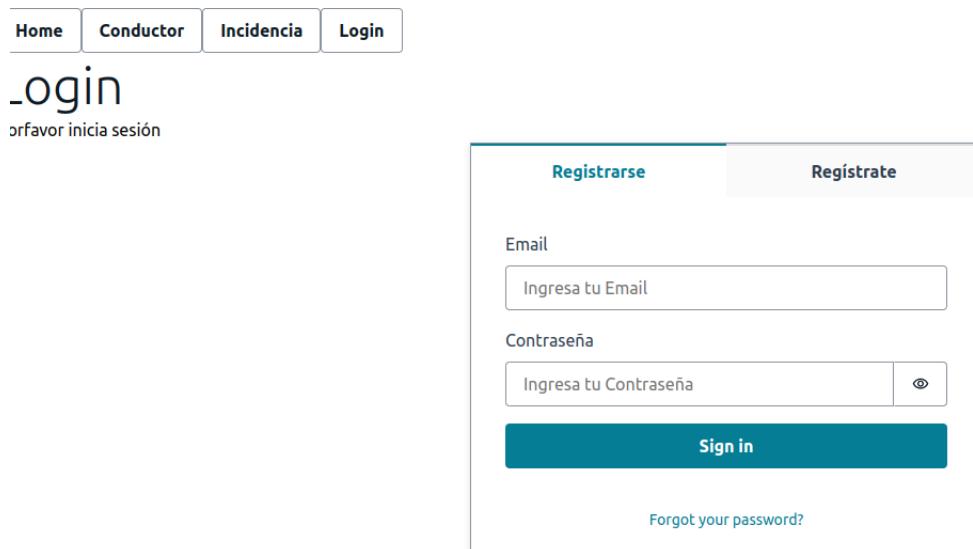


Figura 102: Página de Login

Si intentamos ingresar a las páginas de Conductores, Incidencias o Ubicacion, nos redirigirá a la página de Login, debido a que estas páginas fueron definidas como rutas protegidas. Por lo tanto el usuario debe haber iniciado sesión para poder acceder a las mismas.

- **Path:** /

Descripción: En esta dirección se encontrará la el formulario para poder iniciar sesión o registrarse

- **Path:** /home

Descripción: Esta dirección será la página principal de la aplicación dónde se mostrarán las incidencias más recientes así como una lista de todos los conductores

- **Path:** /conductor/

Descripción: Esta dirección mostrará el perfil del conductor de id correspondiente

- **Path:** /detalle_incidencia

Descripción: Esta dirección mostrará cada incidencia mostrando detalles como hora, fecha, coordenadas

- **Path:** /conductor/id/ubicacion

Descripción: En esta vista se mostrará la ubicación en tiempo real de cada conductor

- **Path:** /conductor/id/incidencias

Descripción: En esta vista se mostrará todas las incidencias registradas por cada conductor

8.3.2. Definición de rutas del backend

La creación de rutas mediante las que el cliente realizará las peticiones y tendrá acceso a las operaciones, así como su funcionamiento en cuanto a obtención de datos y comunicación con el resto de la aplicación.

Para poder utilizar los servicios de Amazon Amplify, es necesario ir al directorio root del proyecto y ejecutar el siguiente comando:

```
npm install -g @aws-amplify/cli
```

copy

Figura 103: Instalación Amplify CLI

Posteriormente, se necesita especificar la región en la cual queremos alojar nuestra aplicación web:

```
Specify the AWS Region
? region: # Your preferred region
Follow the instructions at
https://docs.amplify.aws/cli/start/install/#configure-the-amplify-cli

to complete the user creation in the AWS console
https://console.aws.amazon.com/iamv2/home#/users/create
```

Figura 104: Configuración del proyecto

Utilizando un editor de código, se necesita especificar que estará utilizando el *SDK* de Amazon Amplify:

```
const AWS = require('aws-sdk')
const awsServerlessExpressMiddleware = require('aws-serverless-express/middleware')
const bodyParser = require('body-parser')
const express = require('express')

AWS.config.update({ region: process.env.TABLE_REGION });
```

Figura 105: Implementación del SDK de Amplify

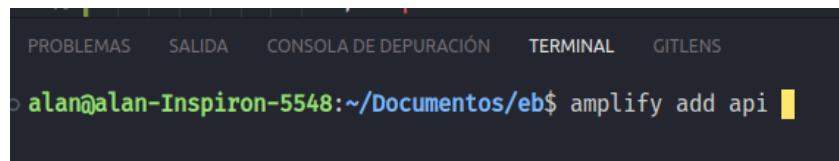
Posteriormente, declaramos una aplicación de ExpressJs la cuál nos permitirá ejecutar entre otras cosas, peticiones HTTP para la comunicación con la base de datos.

```
// declaración de una app de express
const app = express()
app.use(bodyParser.json())
app.use(awsServerlessExpressMiddleware.eventContext())
```

Figura 106: Creación de aplicación de Express

Finalmente, se necesitan definir las rutas de las APIs que se estarán utilizando en el proyecto, las cuales serán dos, la primera se encargará de la aplicación web, y la segunda de realizar la comunicación con el Módulo Central de Procesamiento.

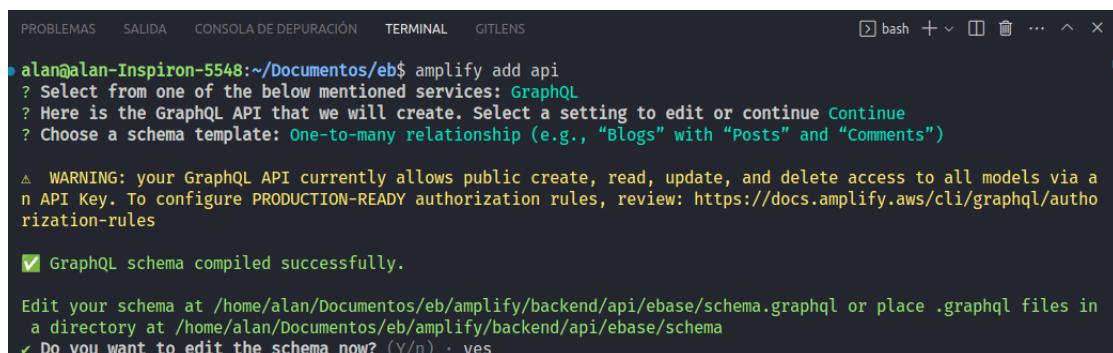
Para agregar una api, se necesita ejecutar el siguiente comando desde el directorio raíz del proyecto.



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add api
```

Figura 107: Amplify add api

La consola de AWS Amplify requiere introducir parámetros con los cuales serán construida nuestra API, para el presente proyecto se estará utilizando una arquitectura mediante GraphQL, por lo cual seleccionaremos dicha opción.



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add api
? Select from one of the below mentioned services: GraphQL
? Here is the GraphQL API that we will create. Select a setting to edit or continue Continue
? Choose a schema template: One-to-many relationship (e.g., "Blogs" with "Posts" and "Comments")
⚠ WARNING: your GraphQL API currently allows public create, read, update, and delete access to all models via a
n API Key. To configure PRODUCTION-READY authorization rules, review: https://docs.amplify.aws/cli/graphql/autho
rization-rules
✓ GraphQL schema compiled successfully.

Edit your schema at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema.graphql or place .graphql files in
a directory at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema
✓ Do you want to edit the schema now? (y/n) · yes
```

Figura 108: Configuración de parámetros de GraphQL

Antes de poder publicar nuestra API en AWS Amplify, se necesita definir el Schema con el cuál se hará el manejo de datos. A continuación se muestran dichos schemas:

```
amplify > backend > api > ebase > schema.graphql
 1 # This "input" configures a global authorization rule to enable
 2 # all models in this schema. Learn more about authorization rules
 3 input AMPLIFY { globalAuthRule: AuthRule = { allow: public } } #
 4
 5 type Conductor @model {
 6   id: ID
 7   nombre: String
 8   apellido: String
 9   incidencias: [Incidencia] @hasMany
10   num_incidencias: Int
11 }
12
13
14 type Incidencia @model {
15   id: ID
16   title: String
17   estado: Boolean
18   conductor: Conductor @belongsTo
19   detalles: [Detalles] @hasOne
20   fecha_hora: Date
21 }
22
23 type Detalles @model {
24   id: ID
25   incidencia: Incidencia @belongsTo
26   ubicacion: String
27   url_video: String
28 }
29
```

Figura 109: Definición de Schemas para el manejo de datos

Finalmente, ejecutaremos el comando *amplify push*, el cuál publicará la API hacáa AWS amplify, generando el endpoint correspondiente a dicha API

Como resultado, tenemos el endpoint de nuestro modelo de GraphQL y nuestra API Key generada por Amplify

The screenshot shows the AWS Amplify CLI interface with the following logs:

```

Deployment completed.
Deploying root stack ebase [ ----- ] 1/3
  amplify-ebase-dev-175126      AWS::CloudFormation::Stack    UPDATE_IN_PROGRESS
  apiebase                      AWS::CloudFormation::Stack    CREATE_IN_PROGRESS
  authebase35f92a6b              AWS::CloudFormation::Stack    UPDATE_COMPLETE
Deployed api ebase [ ===== ] 9/9
  GraphQLAPI                   AWS::AppSync::GraphQLApi   CREATE_COMPLETE
  GraphQLAPINONEDS95A13CF0       AWS::AppSync::DataSource  CREATE_COMPLETE
  GraphQLAPIDefaultApiKey215A6D... AWS::AppSync::ApiKey      CREATE_COMPLETE
  GraphQLAPITransformerSchema3C... AWS::AppSync::GraphQLSchema CREATE_COMPLETE
  Blog                          AWS::CloudFormation::Stack  CREATE_COMPLETE
  Comment                       AWS::CloudFormation::Stack  CREATE_COMPLETE
  Post                          AWS::CloudFormation::Stack  CREATE_COMPLETE
  ConnectionStack               AWS::CloudFormation::Stack  CREATE_COMPLETE
  CustomResourcesjson           AWS::CloudFormation::Stack  CREATE_COMPLETE

✓ Generated GraphQL operations successfully and saved at src/graphql
Deployment state saved successfully.

GraphQL endpoint: [REDACTED].appsync-api.us-east-1.amazonaws.com/graphql
GraphQL API KEY: [REDACTED]jm

GraphQL transformer version: 2

alan@alan-Inspiron-5548:~/Documentos/eb$ 

```

Figura 110: Generación de Endpoint de GraphQL

8.3.3. Creación de la base de datos no relacional

Debido a problemas de integración junto con los servicios de autenticación y de despliegue de Amplify, se decidió utilizar el sistema de gestión de bases de datos DynamoDB. DynamoDB es un servicio de base de datos NoSQL Ofrecido por Amazon Web Services. DynamoDB trabaja con tablas. Estas a su vez, contienen parámetros importantes que se mencionarán a continuación.

- **Primary Key:** Se trata de una clave primaria simple, compuesta por un solo atributo denominado clave de partición. Una clave primaria puede ser una clave de partición o una combinación de clave de partición y clave de ordenación. La clave primaria debe ser única en toda la tabla.
- **Partition Key:** Es la llave principal por la cual se agruparán los datos, y determina cómo se partitiona la información.
- **Sort Key:** Es llave de ordenamiento de los datos.

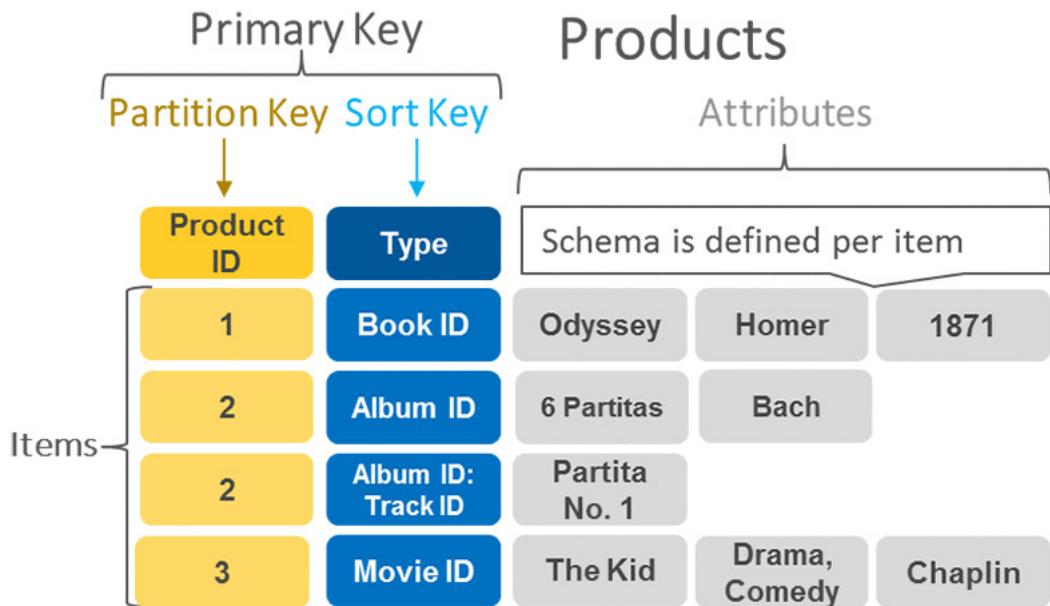


Figura 111: Tablas en DynamoDB

DynamoDB almacena los datos como grupos de atributos, conocidos como elementos. Los elementos son similares a las filas o registros de otros sistemas de bases de datos. DynamoDB almacena y recupera cada elemento en función del valor de la clave principal, que debe ser único.

DynamoDB utiliza el valor de la clave de partición como parámetro de entrada para una función hash interna. El resultado de la función hash determina la partición en la que se almacena el elemento. La ubicación de cada elemento viene determinada por el valor hash de su clave de partición.

Todos los elementos con la misma clave de partición se almacenan juntos y, para las claves de partición compuestas, se ordenan por el valor de la clave de ordenación. DynamoDB divide las particiones por clave de ordenación si el tamaño de la colección crece más de 10 GB[?].

Crear la base de datos en MongoDB.

Para crear nuestras tablas de DynamoDB, se debe ingresar a la consola de AWS.



Figura 112: Consola de AWS

Una vez dentro, ingresamos a la sección del servicio de DynamoDB

Tablas (3) Información						
<input type="checkbox"/>	Nombre	Estado	Clave de partición	Clave de ordenación	Índices	Protección contra eliminaci...
<input type="checkbox"/>	Conductor-trjjwxbd3bbjrjaappwd23ijpl-dev	Activ o	Id (S)	-	0	Desactivada
<input type="checkbox"/>	Detalles-trjjwxbd3bbjrjaappwd23ijpl-dev	Activ o	Id (S)	-	0	Desactivada
<input type="checkbox"/>	Incidencia-trjjwxbd3bbjrjaappwd23ijpl-dev	Activ o	Id (S)	-	1	Desactivada

Figura 113: Tablas generadas mediante los schemas definidos

Como se puede observar, gracias a los pasos realizados en la sección ??, DynamoDB crea automáticamente las tablas creadas en base a los schemas definidos previamente.

8.3.4. Conexión backend con la base de datos

Realizar la conexión de NodeJs con la base de datos MongoDb.

Para realizar la conexión y la integración de los servicios de DynamoDB hacáa nuestra API, se hará uso de AppSync. Las API de GraphQL creadas con AWS AppSync brindan a los desarrolladores frontend la capacidad de consultar varias bases de datos, microservicios y API desde un único punto de conexión de GraphQL.

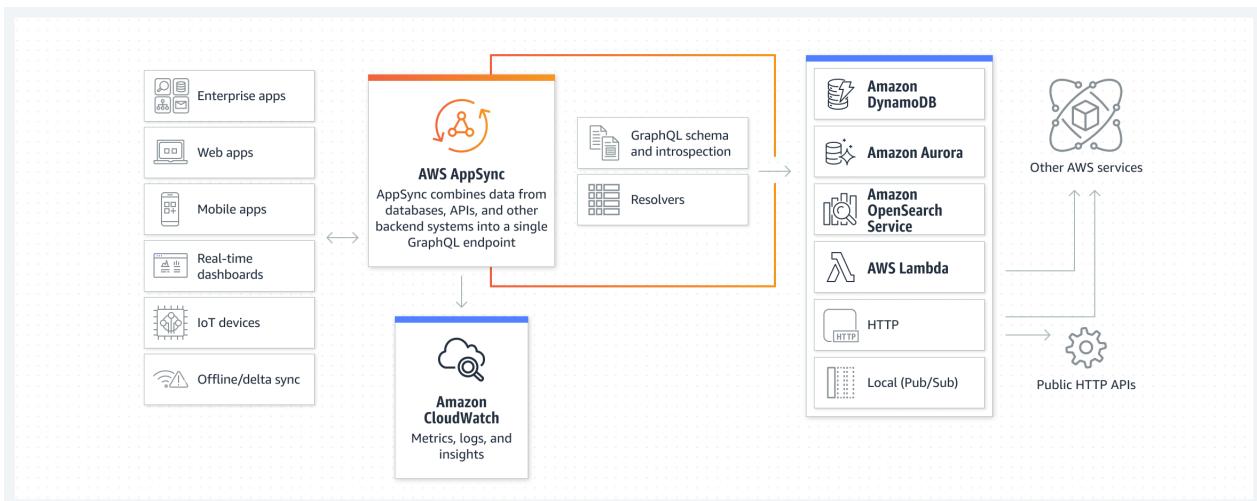


Figura 114: Funcionamiento de Appsnc

AWS AppSync crea las API sin servidor de GraphQL y de publicación o suscripción que simplifican el desarrollo de aplicaciones a través de un único punto de conexión para consultar, actualizar o publicar datos.

Despues de haber realizado los pasos de la sección ??, si ejecutamos el comando *amplify status*, la consola de Amplify nos retornara el endpoint de GraphQL junto con AppSync correspondiente, el cual se utilizará para realizar todas las operaciones con respecto al almacenamiento de datos

```
● alan@alan-Inspiron-5548:~/Documentos/eb$ amplify status

  Current Environment: dev

| Category | Resource name | Operation | Provider plugin   |
|----------|---------------|-----------|-------------------|
| Auth     | ebase35f92a6b | No Change | awscloudformation |
| Api      | ebase         | No Change | awscloudformation |

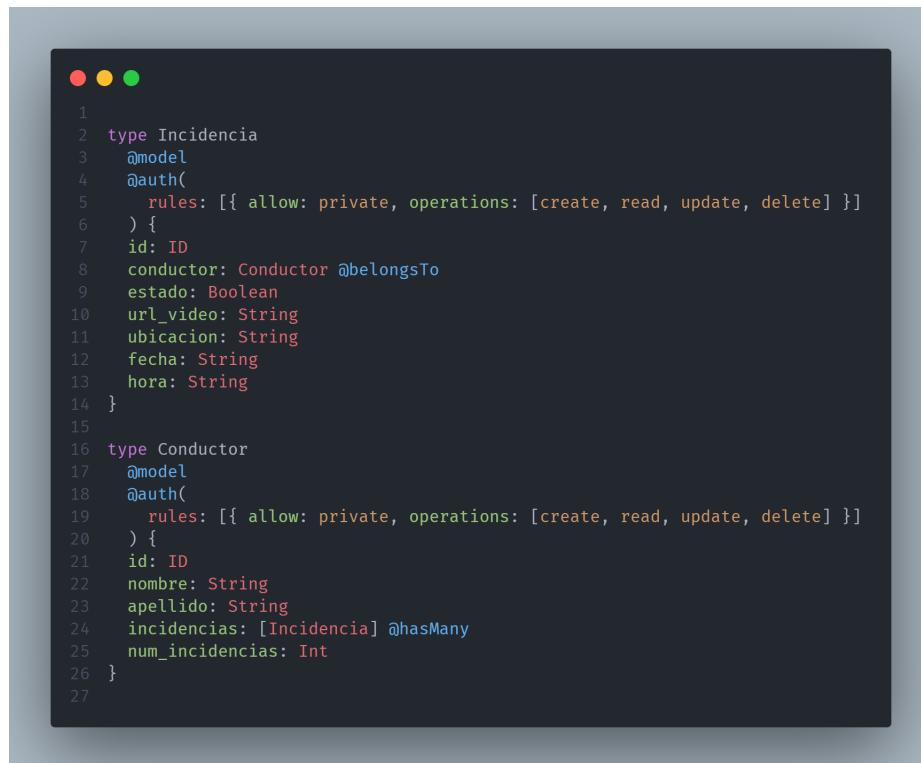


  GraphQL endpoint: https://ckqxuivcobic4rgh72skzudaixy.apsync-a
  GraphQL transformer version: 2
  ○ alan@alan-Inspiron-5548:~/Documentos/eb$
```

Figura 115: Generación de endpoint de GraphQL

8.3.5. Sistema de acceso con credenciales

Definición del Esquema



```
1 type Incidencia
2   @model
3   @auth(
4     rules: [{ allow: private, operations: [create, read, update, delete] }]
5   ) {
6     id: ID!
7     conductor: Conductor @belongsTo
8     estado: Boolean!
9     url_video: String!
10    ubicacion: String!
11    fecha: String!
12    hora: String!
13  }
14}
15
16 type Conductor
17   @model
18   @auth(
19     rules: [{ allow: private, operations: [create, read, update, delete] }]
20   ) {
21     id: ID!
22     nombre: String!
23     apellido: String!
24     incidencias: [Incidencia] @hasMany
25     num_incidencias: Int!
26   }
27
```

Figura 116: Arquitectura Amazon Cognito

Implementación del sistema de acceso con credenciales

Para el sistema de autenticación, se utilizará Amazon Cognito.

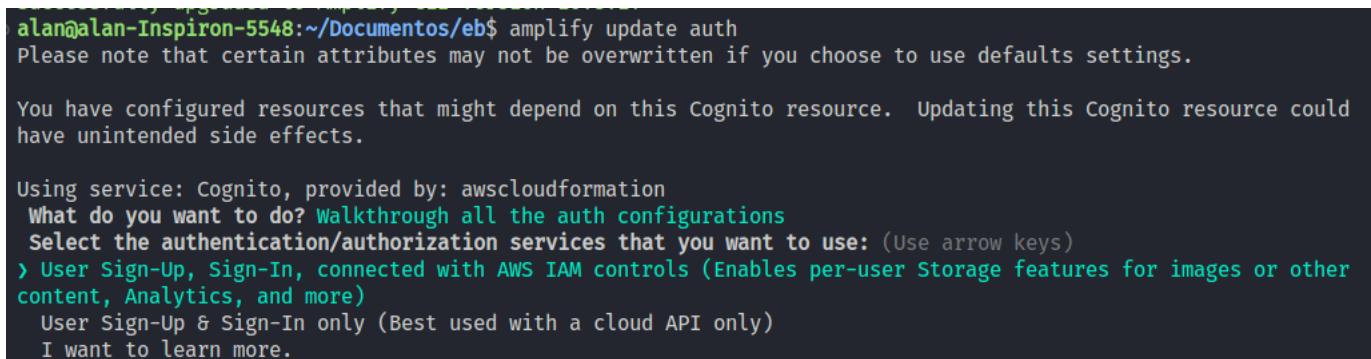
Para poder hacer uso de Amazon Cognito en nuestra aplicación debemos de introducir el siguiente comando:



copy

Figura 117: Implementación de Amazon Cognito en el proyecto

Obteniendo el siguiente menú:



```
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify update auth
Please note that certain attributes may not be overwritten if you choose to use defaults settings.

You have configured resources that might depend on this Cognito resource. Updating this Cognito resource could
have unintended side effects.

Using service: Cognito, provided by: awscloudformation
What do you want to do? Walkthrough all the auth configurations
Select the authentication/authorization services that you want to use: (Use arrow keys)
> User Sign-Up, Sign-In, connected with AWS IAM controls (Enables per-user Storage features for images or other
content, Analytics, and more)
  User Sign-Up & Sign-In only (Best used with a cloud API only)
  I want to learn more.
```

Figura 118: Menú de configuración de Amazon Cognito

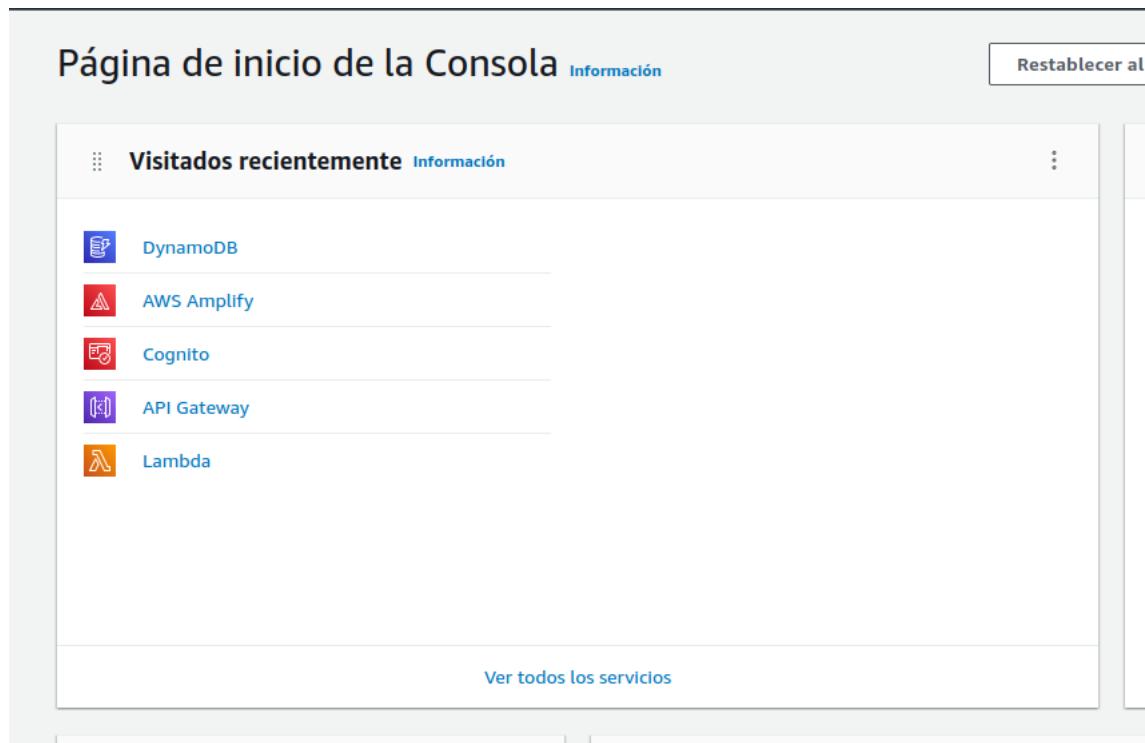


Figura 119: Implementación completada

Entrando a nuestra consola de AWS, en la sección de Amazon Cognito, se puede observar dos grupos de usuarios, uno de tipo Administrador, el cual puede realizar cambios a la configuración de la aplicación, y otro de tipo de Usuario, el cual sólo puede hacer uso del sistema de inicio de sesión y registro ofrecido por Cognito.

Amazon Cognito > Grupos de usuarios				
Grupos de usuarios (2) Información				
Visualice y configure los grupos de usuarios. Estos grupos son directorios de perfiles de usuarios federados y locales. Proporcionan opciones de autenticación para los usuarios.				
	Eliminar	Crear grupo de usuarios		
<input type="text"/> Buscar grupos de usuarios por nombre o ID < 1 > 				
Nombre del grupo de usuarios	ID de grupo de usuarios	Hora de creación		Hora de la actualización más reciente
amplify_backend_manager_dj0gs5937lsru	us-east-1_GzlwWxHQt	Hace 5 días		Hace 5 días
ebase35f92a6b_userpool_35f92a6b-dev	us-east-1_M8QPxoGTw	Hace 7 días		Hace 7 días

Figura 120: Grupos de usuario

Información general sobre el grupo de usuarios

Nombre del grupo de usuarios amplify_backend_manager_dj0gs5937lsru	ARN arn:aws:cognito-idp:us-east-1:387678068467:userpool/us-east-1_GzlwWxHQt	Hora de creación 2 de marzo de 2023, 13:46 GMT-6
ID de grupo de usuarios us-east-1_GzlwWxHQt	Número estimado de usuarios 1	Hora de la actualización más reciente 2 de marzo de 2023, 13:46 GMT-6

▶ Introducción

Usuarios Grupos Experiencia de inicio de sesión Experiencia de inscripción Mensajería Integración de aplicaciones Propiedades del grupo de usuarios

Usuarios (1) Información
Vea, edite y cree usuarios en el grupo de usuarios. Los usuarios habilitados y confirmados podrán iniciar sesión en el grupo de usuarios.

Nombre de usuario	Dirección de correo...	Correo electrónico ...	Estado de la confirmación	Estado
aws-amplify-admin	-	No	Confirmado	Habilitado

Figura 121: Grupos de usuario

Usuarios Grupos Experiencia de inicio de sesión Experiencia de inscripción Mensajería Integración de aplicaciones Propiedades

Usuarios (5) Información
Vea, edite y cree usuarios en el grupo de usuarios. Los usuarios habilitados y confirmados podrán iniciar sesión en el grupo de usuarios.

Nombre de usuario	Dirección de correo elec...	Correo electrónico verifi...	Estado de la confirmación	Estado
19bd77dc-441b-4d3e-8...	maite.diazm98@gmail.com	Sí	Confirmado	Habilitado
473bad4c-b959-45e4-8...	alangam97@gmail.com	No	No confirmado	Habilitado
849ca9f3-6fd3-4750-a8...	mayte_dm@hotmail.com	No	No confirmado	Habilitado
d1df2a48-8374-4fd2-be...	mayte_dm@hotmail.es	No	No confirmado	Habilitado
e6bf3d4f-1807-4325-ba...	alangam97@gmail.com	Sí	Confirmado	Habilitado

Figura 122: Grupos de usuario

8.3.6. Creación de los servicios backend

Para comenzar con el archivo de *Querriesse* tienen las siguientes funciones:

```
export const getConductor = /* GraphQL */ `query GetConductor($id: ID!) {  getConductor(id: $id) {    id    nombre| You, el mes pasado • empez    apellido    incidencias {      items {        id        estado        url_video        ubicacion        fecha_hora        createdAt        updatedAt        conductorIncidentiasId      }    }    nextToken  }  num_incidencias  createdAt  updatedAt}`;
```

Figura 123: Función getConductor

La función de la figura ??, obtiene los datos de un solo Conductor.

```
'export const listConductors = /* GraphQL */ `query ListConductors(  $filter: ModelConductorFilterInput  $limit: Int  $nextToken: String) {  listConductors(filter: $filter, limit: $limit, nextToken: $nextToken) {    items {      id      nombre      apellido      incidencias {        nextToken      }      num_incidencias      createdAt      updatedAt    }    nextToken  }}
```

Figura 124: Función listConductors

La función de la figura 124 obtiene todos los datos de todos los conductores almacenados en la base de datos.

```
export const getIncidencia = /* GraphQL */ `query GetIncidencia($id: ID!) {
  getIncidencia(id: $id) {
    id
    conductor {
      id
      nombre
      apellido
      incidencias {
        nextToken
      }
      num_incidencias
      createdAt
      updatedAt
    }
    estado
    url_video
    ubicacion
    fecha_hora
    createdAt
    updatedAt
    conductorIncidenciasId
  }
};`
```

Figura 125: Función getIncidencia

La función de la figura 125 obtiene los datos de una sola Incidencia.

```
port const listIncidencias = /* GraphQL */ `query ListIncidencias(
  $filter: ModelIncidenciaFilterInput
  $limit: Int
  $nextToken: String
) {
  listIncidencias(filter: $filter, limit: $limit) {
    items {
      id
      conductor {
        id
        nombre
        apellido
        num_incidencias
        createdAt
        updatedAt
      }
      estado
      url_video
      ubicacion
      fecha_hora
      createdAt
      updatedAt
      conductorIncidenciasId
    }
    nextToken
  }
}`
```

Figura 126: Función listIncidencias

La función de la figura 126 obtiene los datos de todas las incidencias de almacenadas en la base de datos.

En cuanto al archivo de *Mutations* se tienen las siguientes funciones:

```
export const createConductor = /* GraphQL */ ` mutation CreateConductor( $input: CreateConductorInput! $condition: ModelConductorConditionInput ) { createConductor(input: $input, condition: $condition) { id nombre apellido incidencias { items { id estado url_video ubicacion fecha_hora createdAt updatedAt conductorIncidenciasId } nextToken } num_incidencias createdAt updatedAt } }`;
```

Figura 127: Función createConductor

La función de la figura 127 se encarga de crear el registro de un conductor en la base de datos.

```
' export const updateConductor = /* GraphQL */ ` mutation UpdateConductor( $input: UpdateConductorInput! $condition: ModelConductorConditionInput ) { updateConductor(input: $input, condition: $condition) { id nombre apellido incidencias { items { id estado url_video ubicacion fecha_hora createdAt updatedAt conductorIncidenciasId } nextToken } num_incidencias createdAt updatedAt } }`;
```

Figura 128: Función updateConductor

La función de la figura 128 se encarga de modificar datos del registro de un conductor.

```
;  
export const deleteConductor = /* GraphQL */ `  
mutation DeleteConductor(  
  $input: DeleteConductorInput!  
  $condition: ModelConductorConditionInput  
) {  
  deleteConductor(input: $input, condition: $condition) {  
    id  
    nombre  
    apellido  
    incidencias {  
      items {  
        id  
        estado  
        url_video  
        ubicacion  
        fecha_hora  
        createdAt  
        updatedAt  
        conductorIncidentiasId  
      }  
      nextToken  
    }  
    num_incidencias  
    createdAt  
    updatedAt  
  }  
}
```

Figura 129: Función deleteConductor

La función de la figura 129 se encarga de eliminar un conductor de la base de datos.

```
;  
export const createIncidencia = /* GraphQL */ `  
mutation CreateIncidencia( You, el mes pasado * empezar  
  $input: CreateIncidenciaInput!  
  $condition: ModelIncidenciaConditionInput  
) {  
  createIncidencia(input: $input, condition: $condition) {  
    id  
    conductor {  
      id  
      nombre  
      apellido  
      incidencias {  
        nextToken  
      }  
      num_incidencias  
      createdAt  
      updatedAt  
    }  
    estado  
    url_video  
    ubicacion  
    fecha_hora  
    createdAt  
    updatedAt  
    conductorIncidentiasId  
  }  
}
```

Figura 130: Función createIncidencia

La función de la figura 130 se encarga de crear una Incidencia en la base de datos.

Para el archivo de *subscriptions* se tienen la siguiente función:

```
export const onCreateIncidencia = /* GraphQL */ `subscription OnCreateIncidencia($filter: ModelSubscriptionIncidenciaFilterInput) {  onCreateIncidencia(filter: $filter) {    id    conductor {      id      nombre      apellido      incidencias {        nextToken      }      num_incidencias      createdAt      updatedAt    }    estado    url_video    ubicacion    fecha_hora    createdAt    updatedAt    conductorIncidenciasId  }}`;
```

Figura 131: Función oncreateIncidencia

La función de la figura 131 se encarga de realizar una consulta cada vez que una Incidencia nueva es dada de alta en la base de datos.

Al estar trabajando con GraphQL dentro del proyecto, la manera en que se podrán realizar las operaciones *CRUD - (Create, Read, Update, Delete)*, será mediante funciones JSON.

Para comprobar que la API permite dichas operaciones, se debe ingresar a la consola de AWS y dirigirse a la sección de AWS AppSync.



Figura 132: Consola de AWS

Posteriormente se necesita ingresar a la sección de consultas.

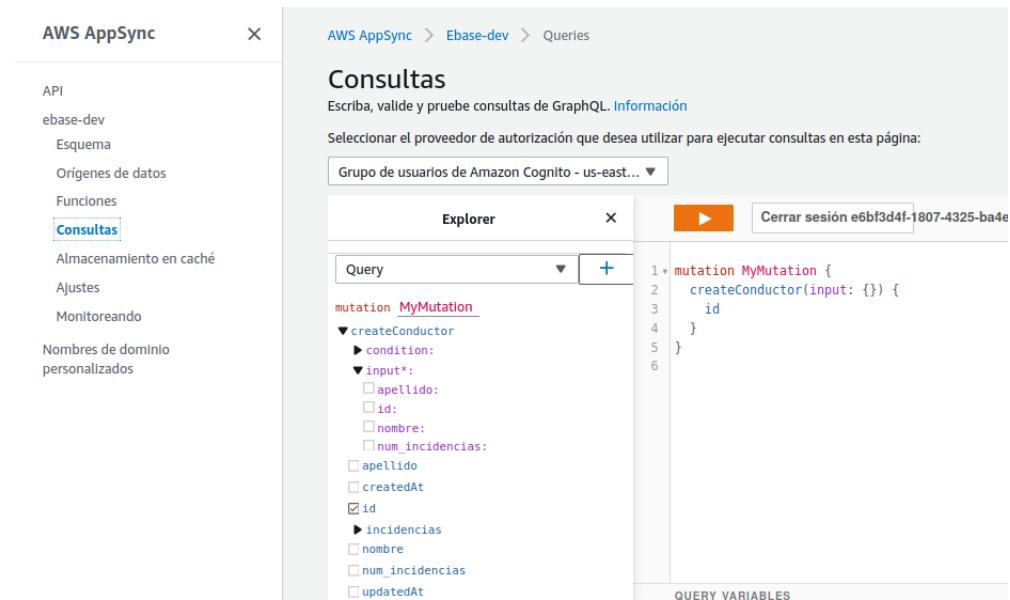


Figura 133: Funcionamiento de AppsSync

AWS permite elegir si realizar un *query*, *mutation*, o *subscription* mediante código JSON

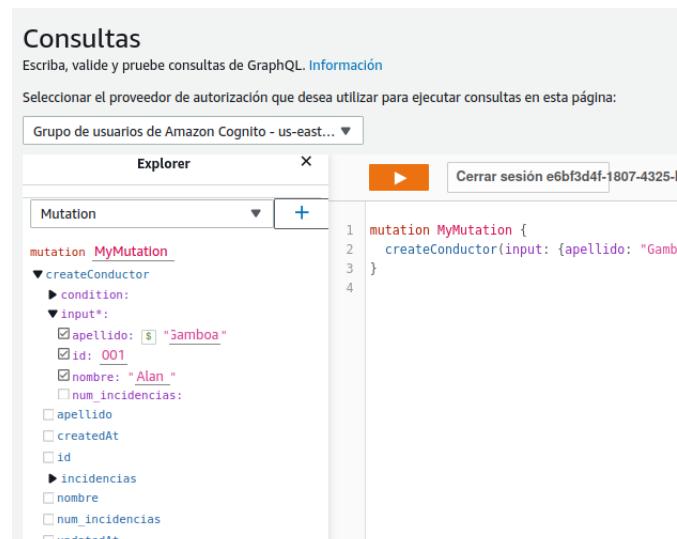


Figura 134: Crear Conductor

En la figura 134 se puede apreciar una sentencia JSON que permite utilizar las funciones previamente creadas para dar de alta un conductor.

8.3.7. Maquetación web

ReactJs

Para el desarrollo del front-end del presente proyecto, se hará uso de la librería de diseño de ReactJs. ReactJs facilita la creación de componentes reutilizables e interactivos para las interfaces de usuario.

Los componentes que darán lugar a las vistas del presente proyecto son los siguientes:

- Layout.jsx: Este componente será la vista principal de la Aplicación Web. Se trata de un diseño tipo *dashboard* que contendrá una sección principal que contendrá etiquetas para poder ingresar a las diferentes vistas de la aplicación. Además estará compuesta también de una sección secundaria que mostrará el contenido de dichas vistas.
- Incidencias.jsx: Este componente se encargará de mostrar todas las incidencias registradas en la base de datos. Las incidencias serán desplegadas en forma de lista.
- Incidencias.jsx: Este componente de mostrar un reporte de incidencia a detalle. Contendrá una ventana que permitirá ver el video del momento de la incidencia registrada. Así como los datos de la fecha y hora. Además de botones para poder confirmar o rechazar la incidencia. Finalmente contendrá el nombre del conductor además de una opción para poder consultar la ubicación en tiempo real del conductor.
- Ubicación.jsx: Este componente mostrará la ubicación en tiempo real del conductor con ayuda del servicio de diseño de mapas Leaflet.
- Conductores.jsx: Este componente mostrará todos los conductores registrados en la base de datos en forma de lista.

De acuerdo con los componentes explicados anteriormente, las vistas que contendrá la aplicación web son las siguientes:

- Página Principal

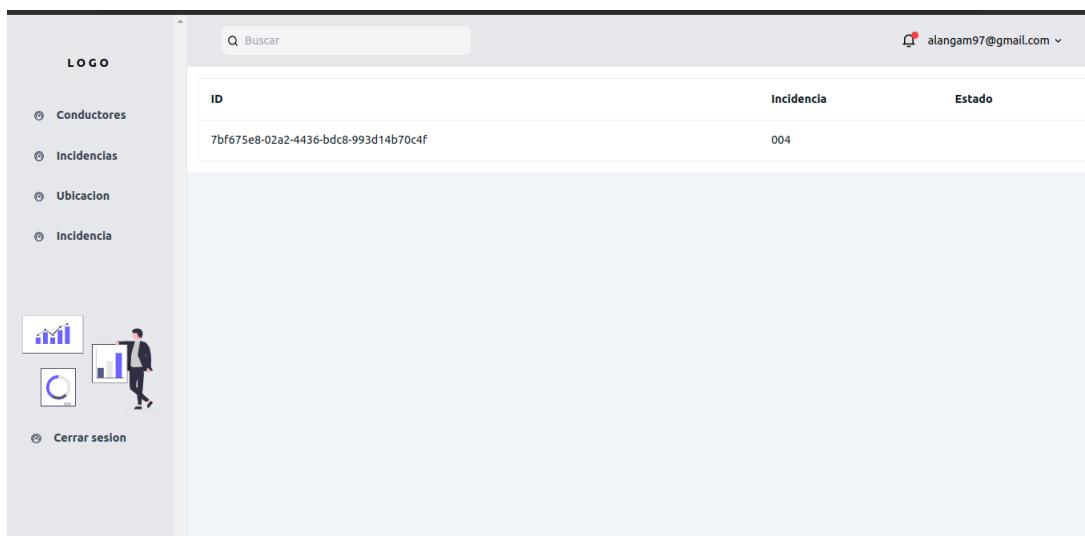


Figura 135: Página Principal - Layout.jsx

- Reporte de Incidencia

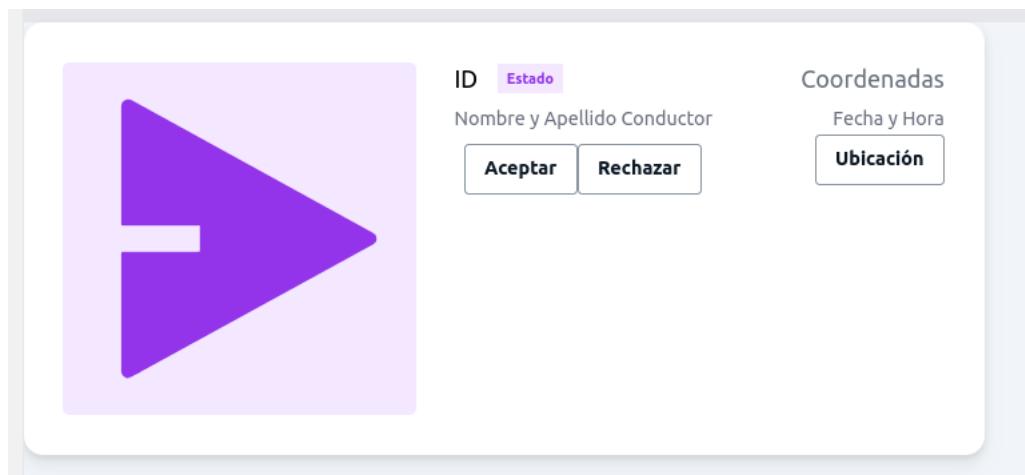


Figura 136: Vista Reporte Incidencia Incidencia - Incidencia.jsx

- Ubicación

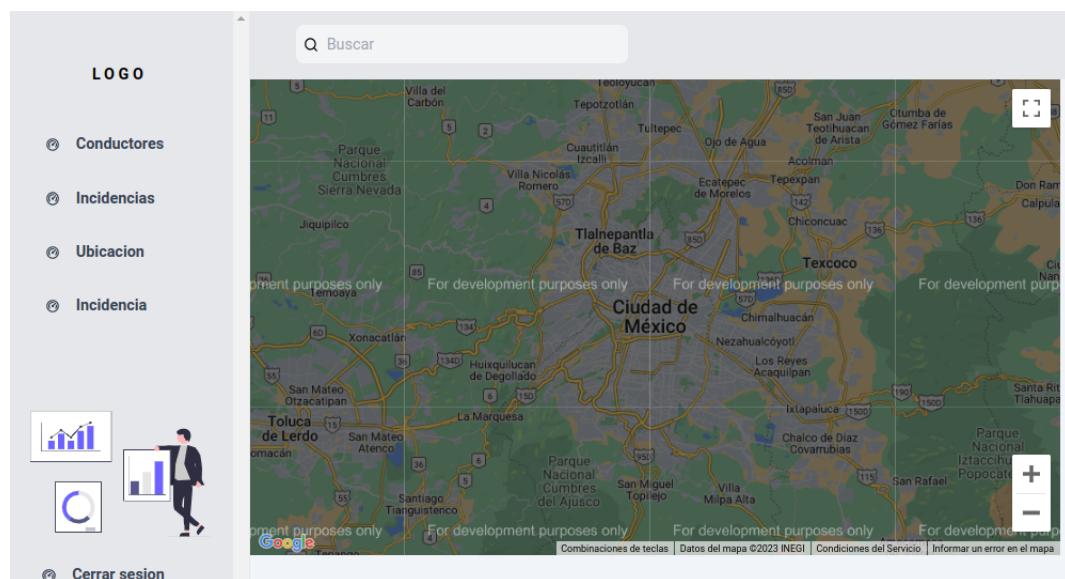


Figura 137: Vista Ubicacion

- Conductores

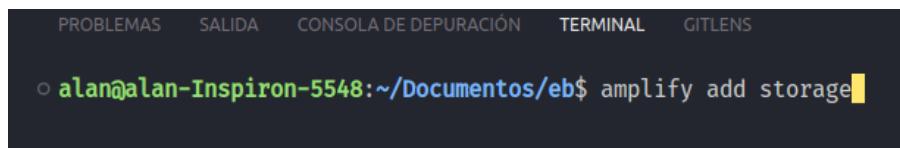
The screenshot shows a web-based application interface. On the left, there is a sidebar titled "LOGO" containing links for "Conductores", "Incidencias", "Ubicacion", and "Incidencia". Below these are three icons: a bar chart, a map, and a person walking. At the bottom of the sidebar is a link for "Cerrar sesion". The main area features a search bar at the top right with the placeholder "Buscar" and a user icon with the email "alangam97@gmail.com". A table lists a single conductor entry:

ID	Conductor	Fecha	Acciones
001	Alan	2023-04-18T00:39:17.446Z	<button>Modificar</button> <button>Eliminar</button>

Figura 138: Vista Conductores - Conductores.jsx

8.3.8. Enlace de Amazon S3 con el sistema backend

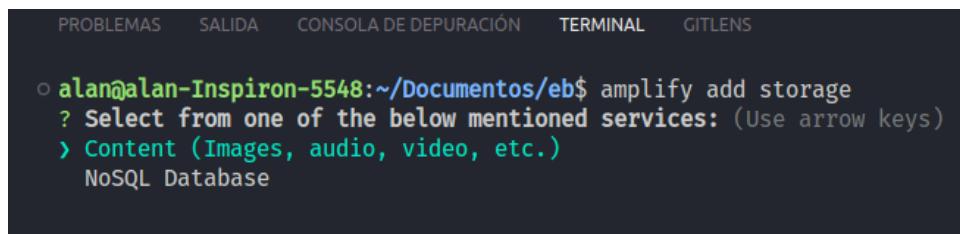
Utilizando un editor de código, y desde el directorio raíz de la aplicación, se deberá introducir el siguiente comando:



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
○ alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
```

Figura 139: Configuración de servicio de almacenamiento S3

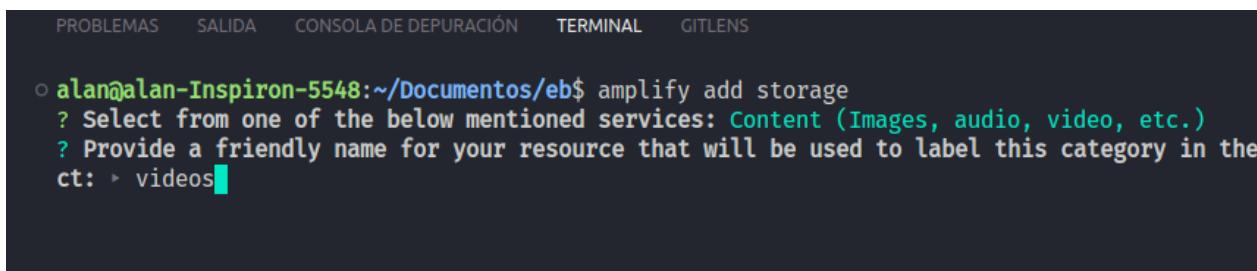
Posteriormente, se requiere especificar que tipo de servicio de almacenamiento se integrará a la aplicación (multimedia o base de datos NoSQL). Para el presente proyecto, se utilizará el almacenamiento de contenido multimedia, por lo tanto, se seleccionará dicha opción.



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
○ alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: (Use arrow keys)
> Content (Images, audio, video, etc.)
NoSQL Database
```

Figura 140: Generación de Endpoint de GraphQL

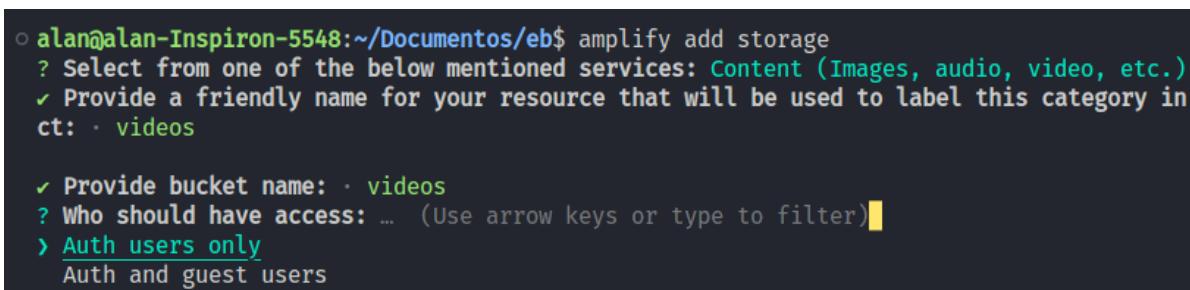
Ingresamos el nombre de nuestro espacio de almacenamiento:



```
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL GITLENS
○ alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
? Provide a friendly name for your resource that will be used to label this category in the
ct: > videos
```

Figura 141: Generación de Endpoint de GraphQL

Después, se necesita establecer cuantos usuarios, así como cuales podrán acceder a dicho servicio:



```
○ alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Provide a friendly name for your resource that will be used to label this category in the
ct: > videos
✓ Provide bucket name: > videos
? Who should have access: ... (Use arrow keys or type to filter)
> Auth users only
Auth and guest users
```

Figura 142: Generación de Endpoint de GraphQL

```

○ alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Provide a friendly name for your resource that will be used to label this category in
ct: · videos

✓ Provide bucket name: · videos
✓ Who should have access: · Auth and guest users
? What kind of access do you want for Authenticated users? ... (Use arrow keys or type to
  ● create/update
  ● read
  >● delete
(Use <space> to select, <ctrl + a> to toggle all)

```

Figura 143: Generación de Endpoint de GraphQL

```

? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Who should have access: · Auth and guest users
✓ What kind of access do you want for Authenticated users? · create/update, read, delete
? What kind of access do you want for Guest users? ... (Use arrow keys or type to filter)
  >● create/update
  ● read
  ○ delete
(Use <space> to select, <ctrl + a> to toggle all)

```

Figura 144: Generación de Endpoint de GraphQL

```

Edit your schema at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema.graphql or plac
raphql files in a directory at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema
✓ Successfully pulled backend environment dev from the cloud.

```

Current Environment: dev

Category	Resource name	Operation	Provider plugin
Storage	videos	Create	awscloudformation
Auth	ebase35f92a6b	Update	awscloudformation
Function	S3Trigger6956e03e	No Change	awscloudformation
Api	ebase	No Change	awscloudformation

? Are you sure you want to continue? (Y/n) >

Figura 145: Generación de Endpoint de GraphQL

Al ingresar a la consola de servicios de AWS, en la sección de buckets de S3, se puede observar que se encuentra el bucket recién creado llamado *videos175126-dev*.

Buckets (3) Info					
Los buckets son contenedores de datos almacenados en S3. Más información					
	Copiar ARN	Vaciar	Eliminar	Crear bucket	
C	Copiar ARN	Vaciar	Eliminar	Crear bucket	
<input type="text"/> Buscar buckets por nombre					
Nombre	▲	Región de AWS	▼	Acceso	▼
amplify-ebase-dev-175126-deployment		EE. UU. Este (Norte de Virginia) us-east-1		Los objetos pueden ser públicos	28 Feb 2023 5
ebase6c6dcb3b214e4c3fa82df5d47dce7c22175126-dev		EE. UU. Este (Norte de Virginia) us-east-1		Los objetos pueden ser públicos	25 Mar 2023 1
videos175126-dev		EE. UU. Este (Norte de Virginia) us-east-1		Los objetos pueden ser públicos	17 Apr 2023 1

Figura 146: Tablas generadas mediante los schemas definidos

8.3.9. Despliegue de la aplicación web

Para poder alojar y desplegar la aplicación web en un servicio de *hosting*, se utilizará Amplify Hosting [53].

Se necesita ingresar a la consola de AWS y seleccionar el servicio de Amplify Hosting

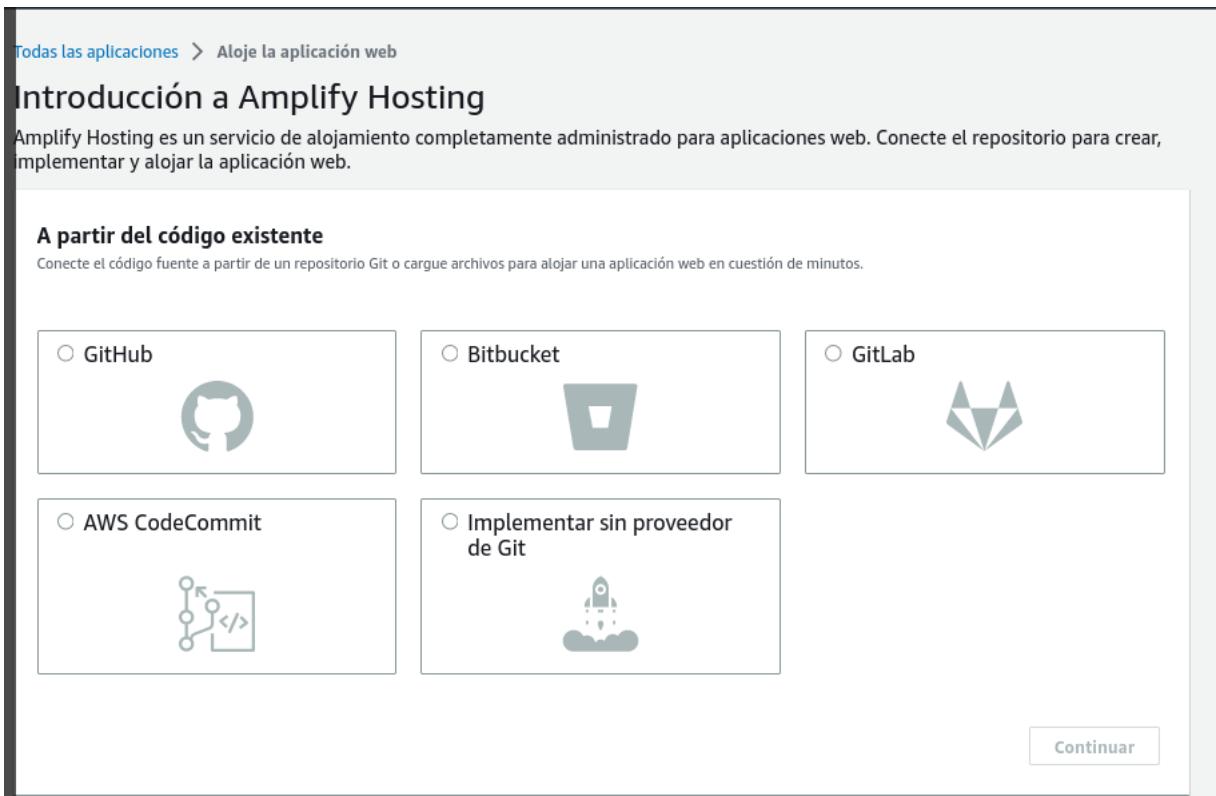


Figura 147: Amplify Hosting

Hosting requiere alguna fuente de alojamiento de código para poder desplegar la aplicación web. En este caso, la aplicación web fue alojada utilizando *Github*, por lo tanto, se selecciona esta opción. Posteriormente, se requiere seleccionar el repositorio el cual será alojado, en este caso, el nombre de dicho repositorio es *eb*. A su vez, se selecciona la ramificación *main* para ser desplegada.

Agregar ramificación de repositorio

Se autorizó a GitHub correctamente.

Proveedor de servicios de repositorio

GitHub

Repositorios actualizados recientemente

alangamboa97/eb

Ver permisos de GitHub

Si no ve el repositorio en el menú desplegable de arriba, asegúrese de que la aplicación Amplify de GitHub tenga permisos en el repositorio. Si su repositorio todavía no aparece, envíe una confirmación y haga clic en el botón de actualización.

Ramificación

Seleccione una ramificación del repositorio.

main

Figura 148: ramificación de repositorio de Github

Resultados Acceso a la pagina web desde el buscador con la siguiente URL:
<https://main.dcst1c83llut3.amplifyapp.com/login>

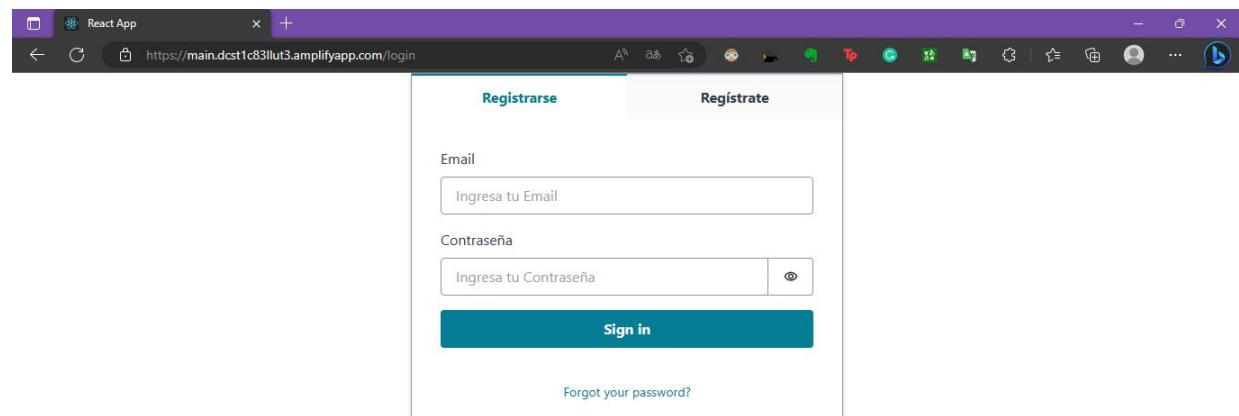


Figura 149: Acceso a la página web desde el navegador

9. Pruebas

9.1. Módulo de Visión Artificial

9.1.1. Prueba de detección y clasificación de somnolencia:

- Verificar la capacidad de `dlib` para la detección de rostros, ojos y boca en diferentes condiciones de iluminación y fondos.
- Simular estados de apertura y cierre de ojos y boca para confirmar la precisión en la clasificación de la somnolencia.
- Evaluar la respuesta del sistema ante cambios rápidos en los estados faciales.

9.2. Módulo de Comunicaciones

9.2.1. Prueba de Conectividad LTE:

- Verificar la estabilidad y velocidad de la conexión LTE para la transferencia de información de posicionamiento GPS.
- Evaluar la capacidad de recuperación en casos de pérdida temporal de conectividad.

9.2.2. Prueba de Posicionamiento GPS:

- Confirmar la precisión del posicionamiento proporcionado por GNSS en diferentes entornos geográficos.

9.3. Estación Base (Amazon Amplify)

9.3.1. Prueba de Integración con GitHub:

- Confirmar la sincronización automática entre el repositorio en GitHub y la aplicación web en Amplify Hosting.
- Evaluar la eficacia del ciclo de desarrollo y despliegue.

9.4. Gestión de Contenido Multimedia y Seguridad

9.4.1. Prueba de Almacenamiento en S3:

- Evaluar la velocidad de acceso y la escalabilidad del almacenamiento en la nube S3 para contenido multimedia.
- Verificar la disponibilidad y confiabilidad de los archivos almacenados.

9.4.2. Prueba de Autenticación y Gestión de Identidades (Amazon Cognito):

- Validar la seguridad y eficacia del proceso de autenticación de usuarios.
- Evaluar la gestión de permisos y credenciales de acceso.

9.5. Base de Datos (DynamoDB)

9.5.1. Prueba de Rendimiento de DynamoDB:

- Evaluar la velocidad de lectura y escritura de datos en DynamoDB bajo carga variable.
- Confirmar la escalabilidad del sistema de gestión de bases de datos.

9.6. Aplicación Web (Amplify y React)

9.6.1. Prueba de Rendimiento del Backend:

- Evaluar la capacidad de respuesta y eficiencia del backend.
- Verificar la gestión adecuada de las solicitudes concurrentes.

9.6.2. Prueba de Interfaz de Usuario (Frontend React):

- Evaluar la velocidad de carga y la reactividad de la interfaz de usuario en diferentes dispositivos y navegadores.

9.7. Pruebas de Integración del Sistema

9.7.1. Prueba de Interacción entre Módulos:

- Verificar la correcta integración y comunicación entre el módulo de visión artificial, la estación base y la base de datos.

9.7.2. Prueba de Alertas y Monitoreo en Tiempo Real:

- Confirmar la emisión de alertas ante la detección de síntomas de somnolencia.
- Evaluar la visualización en tiempo real de la ubicación del conductor desde la aplicación web.

9.8. Evaluación del Sistema Final

9.8.1. Prueba de Funcionalidad Integral:

- Realizar una prueba completa del sistema para validar su funcionalidad global en condiciones simuladas y reales.

9.8.2. Prueba de Resiliencia del Sistema:

- Evaluar la capacidad del sistema para recuperarse de posibles fallas o interrupciones en los servicios y módulos.

9.8.3. Prueba de Seguridad Integral:

- Realizar análisis de seguridad para identificar posibles vulnerabilidades en la comunicación, autenticación y almacenamiento de datos.

9.8.4. Prueba de Usabilidad:

- Obtener retroalimentación sobre la experiencia del usuario en la aplicación web y la interacción con las alertas.

9.9. Resultados y Ajustes:

- Evaluar los resultados de cada prueba y realizar ajustes en el diseño o implementación según sea necesario.
- Utilizar los resultados como base para iteraciones adicionales en el desarrollo, siguiendo el enfoque de la metodología en espiral.

10. Validación

10.0.1. Resultados

11. Conclusion y aportaciones

Con base en los objetivos planteados, se llevó a cabo el diseño y análisis del sistema utilizando la metodología en espiral, que implica un enfoque iterativo y progresivo en el desarrollo. El sistema se dividió en módulos y submódulos, permitiendo un enfoque más detallado en cada uno de ellos. Esta aproximación ayudó a definir los elementos y algoritmos necesarios para la implementación del sistema. En este trabajo, se presentan los resultados de las investigaciones que para la elección de los elementos principales del sistema y, posteriormente, los resultados del diseño de cada módulo propuesto.

En el Submódulo de Visión Artificial, se logró la detección del rostro, los ojos y la boca, seguida de la clasificación de la somnolencia. Este proceso consideró la apertura de los ojos y la boca, así como la duración de dichos estados. La utilización de dlib para la detección facial y la relación MOR y EOR a través de puntos de referencia faciales contribuyeron a la implementación para el reconocimiento de signos de somnolencia.

En cuanto a la comunicación y transferencia de archivos, se optó por el módulo SIM7600G para Jetson Nano, que integra GNSS para el posicionamiento GPS. La conectividad LTE fue seleccionada, y se configuró un intervalo de tiempo de 10 segundos para el envío de información de posicionamiento GPS.

En el proceso de desarrollo de la Estación Base, se tomó la decisión estratégica de implementar la suite de herramientas proporcionada por Amazon Amplify. Esta elección se fundamentó en la eficacia y versatilidad que ofrece Amplify Hosting, al mismo tiempo que permite una integración fluida con GitHub. De esta manera, los cambios realizados en el repositorio se reflejan de manera automática en la aplicación web, agilizando el ciclo de desarrollo y asegurando una sincronización eficiente entre el código fuente y la implementación en producción.

Para abordar la gestión de contenido multimedia, se optó por el servicio de almacenamiento en la nube S3 de Amazon Web Services (AWS). Esta solución proporciona una infraestructura escalable y segura para el almacenamiento de archivos, garantizando un acceso rápido y confiable a los recursos multimedia necesarios para la aplicación.

En cuanto a la seguridad y gestión de credenciales, la responsabilidad recayó en Amazon Cognito. Este servicio facilita la autenticación de usuarios y la gestión de identidades de manera segura, ofreciendo un entorno robusto para la administración de permisos y credenciales de acceso. La elección de Amazon Cognito contribuye a garantizar la integridad y confidencialidad de la información sensible manejada por la Estación Base.

Para el manejo eficiente de la base de datos, se eligió DynamoDB como el sistema de gestión de bases de datos. La elección se basó en su sólida compatibilidad con las herramientas de AWS y su capacidad para integrarse sin problemas en el ecosistema de servicios en la nube.

En lo que respecta a la aplicación web, se implementaron tecnologías líderes en el desarrollo web moderno. Node.js fue seleccionado para el desarrollo del backend, proporcionando un entorno de ejecución eficiente y orientado a eventos. Por otro lado, React se utilizó en el frontend, permitiendo

la construcción de interfaces de usuario dinámicas y reactivas. Esta combinación de tecnologías garantiza un rendimiento óptimo, así como una experiencia de usuario fluida y atractiva.

La implementación del sistema se llevó a cabo de manera iterativa, siguiendo los principios de la metodología en espiral. En cada fase del desarrollo, se llevaron a cabo pruebas y validaciones para asegurar la funcionalidad del sistema. Los resultados obtenidos de estas pruebas fueron evaluados y utilizados como base para realizar ajustes en el diseño como en la implementación.

El sistema final permitió detectar síntomas de somnolencia y emitir alertas al conductor. Además, proporciona la ubicación del conductor en tiempo real, permitiendo el monitoreo desde la aplicación web asociada. Esta aplicación desempeña un papel integral al gestionar la información de los conductores y supervisar los reportes de incidencias enviados desde el módulo central de procesamiento. La interacción entre la infraestructura de AWS y las tecnologías Node.js y React en el frontend y backend, respectivamente, contribuyó significativamente a la implementación y resiliencia del sistema en conjunto.

12. Referencias

Referencias

- [1] B. W. Boehm, “A spiral model of software development and enhancement,” *Computer*, vol. 21, no. 5, pp. 61–72, 1988.
- [2] NVIDIA, “Nvidia jetson nano,” Disponible en: <https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-nano/>, consultado el 26 de enero de 2022.
- [3] N. Bhat. (s/f) Content delivery network. Disponible en: <https://learn.microsoft.com/es-es/power-apps/maker/portals/configure/configure-cdn>. Consultado el 26 de enero de 2022.
- [4] nPerf, “Cobertura 3g / 4g / 5g telcel mobile,” Disponible en: <https://www.nperf.com/es/map/MX/-/2004799.Telcel-Mobile/signal/>, accedido el 22 de diciembre de 2022.
- [5] W. H. Organization. (2018) Global status report on road safety 2018. Disponible en: <http://www.who.int/publications/i/item/9789241565684>. Consultado el 26 de enero de 2022.
- [6] A. E. Paez Mario. (2017) Herramientas para la seguridad en la movilidad, modelos predictivos de somnolencia en conductores. Disponible en: <https://imt.mx/resumen-boletines.html?IdArticulo=449&IdBoletin=168>. Consultado el 26 de enero de 2022.
- [7] Beetrack. (s/f) Rastreo satelital de vehiculos: ¿quÃ© es mejor que el gps tradicional? Disponible en: <https://www.beetrack.com/es/blog/rastreo-satelital-de-vehiculos>. Consultado el 26 de enero de 2022.
- [8] J. Chase. (2021) The evolution of internet of things. Disponible en: <https://pure.coventry.ac.uk/ws/portalfiles/portal/40893306/Binder3.pdf>. Consultado el 26 de enero de 2022.
- [9] G. M. Agustina, S. J. D., C. J. A., and C. W. H., “Sistemas de detección de somnolencia en conductores inicio desarrollo y futuro,” *Revista Ingenieria y Region*, vol. 13, no. 1, pp. 159–168, 2015. [Online]. Available: <https://journalusco.edu.co/index.php/iregion/article/view/717/1372>
- [10] V. E. R. A., “Influencia de un sistema de geolocalizacion en el control y monitoreo de vehiculos con dispositivos gps en una empresa logistica,” 2017, <https://repositorio.ucv.edu.pe/bitstream/handle/20.500.12692/17105/Vilca.ERA.pdf?sequence=1&isAllowed=y>.
- [11] D. W. R. Roehrs T, Carskadon MA, “T. daytime sleepiness and alertness,” Disponible en: https://www.researchgate.net/publication/312502433_Daytime_Sleepiness_and_Alertness, 2017, consultado el 26 de enero de 2022.
- [12] IBM, “Como funciona la vision artificial,” Disponible en: <https://www.ibm.com/mx-es/topics/computer-vision>, s/f, consultado el 26 de enero de 2022.
- [13] aws. (s/f) ¿quÃ© es el aprendizaje profundo? Disponible en: <https://aws.amazon.com/es/what-is/deep-learning/>. Consultado el 26 de enero de 2022.

- [14] AWS. (2023) QuÃ© es una cdn. Disponible en: <https://aws.amazon.com/es/what-is/cdn/>. Consultado el 26 de enero de 2022.
- [15] M. P. and Grance, "The nist definition of cloud computing," NIST special publication, Tech. Rep., 2011.
- [16] Tarify.Win! (s/f) Red wwan que es, para que sirve y como funcionan. Disponible en: <https://tarify.win/definiciones/red-wwan/>. Consultado el 26 de enero de 2022.
- [17] A. La Rosa. (2021) Lpwan como base de comunicaciones para iot. Disponible en: <https://pandorafms.com/blog/es/que-eslpwan/?msclkid=08006fc5cf4611ecbc00aa3b275ed2b5>. Consultado el 8 de mayo de 2022.
- [18] A. S. Gillis. (s/f) Lte (long-term evolution). Disponible en: [https://www.techtarget.com/searchmobilecomputing/definition/Long-Term-Evolution-LTE#:~:text=LTE%20\(Long%2DTerm%20Evolution\)%20is%20a%20fourth%2Dgeneration,%2Dgeneration%20\(3G\)%20technology](https://www.techtarget.com/searchmobilecomputing/definition/Long-Term-Evolution-LTE#:~:text=LTE%20(Long%2DTerm%20Evolution)%20is%20a%20fourth%2Dgeneration,%2Dgeneration%20(3G)%20technology). Consultado el 8 de mayo de 2022.
- [19] G. Cornetta, "Compromiso entre modulacion y codificacion."
- [20] E. Fombona Cadavieco, J. y VÃ¡zquez-Cano. (2017) Posibilidades de utilizacion de la geocalizacion y realidad aumentada en el ambito educativo. Disponible en: <https://www.redalyc.org/pdf/706/70651145014.pdf>. Consultado el 26 de enero de 2022.
- [21] P. S., B. N., D. A., D. V., and B. N. (2020) Driver drowsiness detection using machine learning with visual behaviour. Disponible en: <https://ijcrt.org/papers/IJCRT2006408.pdf>. Consultado el 8 de mayo de 2022.
- [22] K. V. Kanodia G. and M. G. (2020) Detection of drowsiness and distraction of drivers using cnn. Disponible en: <https://www.trendytechjournals.com/files/issues/volume4/issue7-5.pdf>. Consultado el 8 de mayo de 2022.
- [23] S. M. K. and A. T., "Driver drowsiness detection system using convolutional neural networks." Disponible en: <https://ijarsct.co.in/Paper3399.pdf>, 2020, consultado el 8 de mayo de 2022.
- [24] C.-M. Ivask, "Raspberry pi based system for visual object detection and tracking," Disponible en: https://a-lab.ee/edu/sites/default/files/Ivask_BSc.pdf, 2015, consultado el 8 de mayo de 2022.
- [25] R. A. and L. D. J., "Diseno e implementacion de sistema de vision artificial para alerta y deteccion de somnolencia mediante aprendizaje profundo aplicable en conductores de vehiculos," Disponible en: <https://repositorioslatinoamericanos.uchile.cl/handle/2250/4756017>, 2021, consultado el 8 de mayo de 2022.
- [26] L. Gallardo Gomez, "Aplicacion de vision por computador y machine learning al guiado de un robot movil basado en raspberry pi," Disponible en: <https://idus.us.es/handle/11441/127072>, 2021, consultado el 8 de mayo de 2022.
- [27] N. JimÃ©nez Varela, "Evaluacion de la plataforma nvidia jetson nano para aplicaciones de vision artificial," Disponible en: <https://repositorio.uam.es/handle/10486/698381>, 2021, consultado el 8 de mayo de 2022.

- [28] J. R. Elektrika, “Comparative study of computer vision based line followers using raspberry pi and jetson nano,” Disponible en: https://www.researchgate.net/publication/357545530_Comparative_Study_of_Computer_Vision_Based_Line_Followers_Using_Raspberry_Pi_and_Jetson_Nano, 2021, consultado el 8 de mayo de 2022.
- [29] M. Limaquispe and E. Sánchez, “Sistema de detección de somnolencia mediante inteligencia artificial en conductores de vehículos para alertar la ocurrencia de accidentes de tránsito,” Disponible en: <http://repositorio.unh.edu.pe/handle/UNH/2327>, 2018, consultado el 8 de mayo de 2022.
- [30] A.-G. P. K. Eugenio, B. L. H. and P. I. J. Alejandro, “Diseno de un sistema electronico para detectar somnolencia en automovilistas por medio de la actividad ocular,” Disponible en: <https://tesis.ipn.mx/handle/123456789/27287>, 2019, consultado el 26 de enero de 2022.
- [31] R. A. M. Becerril, E. L. and E. T. J. Velázquez, “Sistema para la detección del estado de somnolencia en seres humanos, con reconocimiento de patrones,” Disponible en: https://uptexcoco.edomex.gob.mx/sites/uptexcoco.edomex.gob.mx/files/files/2020/articulos-Tesis/sistema_somnolencia.pdf, s/f, consultado el 26 de enero de 2022.
- [32] H. F. Solás, “Sistema de detección de somnolencia.” Disponible en: <https://riull.ull.es/xmlui/bitstream/handle/915/30312/Sistema%20de%20deteccion%20de%20somnolencia.pdf?sequence=1&isAllowed=y>, 2022, consultado el 26 de enero de 2022.
- [33] C. Pacay and B. Fanny., “Desarrollar un prototipo de reconocimiento facial basado en machine learning para detectar estado de somnolencia en conductores de una cooperativa de transporte,” Disponible en: <http://repositorio.ug.edu.ec/handle/redug/49449?mode=full>, 2020, consultado el 26 de enero de 2022.
- [34] E. L. Benavides and M. M. Medina., “Sistema basado en la detección y notificación de somnolencia para conductores de autos,” Disponible en: <https://repositorio.unicordoba.edu.co/bitstream/handle/ucordoba/536/Trabajo%20de%20grado.pdf?sequence=1&isAllowed=y>, s/f, consultado el 26 de enero de 2022.
- [35] Á. H. Adrian, “Detección de somnolencia para conducción sin accidentes,” Disponible en: <http://hdl.handle.net/10045/124695>, 2022, consultado el 26 de enero de 2022.
- [36] G. Vilca and J. Eduardo, “Diseno e implementacion de un sistema de geolocalizacion en interiores para plataforma android via la red enterprise wlan de la pucp,” Disponible en: <https://tesis.pucp.edu.pe/repositorio/handle/20.500.12404/7156>, 2016, consultado el 26 de enero de 2022.
- [37] C. E. E. Ramírez, R. E. R. Flores, and J. A. V. Hernández, “Propuesta de un sistema de geolocalización y monitoreo via gps/gsm/gprs aplicado a un pulsómetro para personas con enfermedades vasculares,” Disponible en: <https://tesis.ipn.mx/handle/123456789/27690>, 2018, consultado el 26 de enero de 2022.
- [38] B. R. A. V. Espinoza, “Influencia de un sistema de geolocalización en el control y monitoreo de vehículos con dispositivos gps en una empresa logística,” Disponible en: , 2017, consultado el 26 de enero de 2022.

- [39] A. J. Stephahn Baller, "Deepedgebench: Benchmarking deep neural networks on edge devices," Disponible en: <https://arxiv.org/abs/2108.09457>, accedido el 22 de diciembre de 2022.
- [40] I. Kalb, *Object-Oriented Python*, 1st ed. No Starch-Press, 2021.
- [41] A. con Alf, "La libreria numpy," Disponible en: <https://aprendeconalf.es/docencia/python/manual/numpy/>, accedido el 22 de diciembre de 2022.
- [42] M. Cid, "Categorias lte o 4g: que son y que velocidades maximas ofrece cada una de ellas," Disponible en: <https://www.xatakamovil.com/conectividad/categorias-lte-o-4g-que-son-y-que-velocidades-maximas-ofrece-cada-una-de-ellas>, accedido el 22 de diciembre de 2022.
- [43] ADNPolitico, "¡No aceleres! estos son los nuevos lÃmites de velocidad 2022 en mÃ©jico," Disponible en: <https://politica.expansion.mx/mexico/2022/05/19/estos-son-nuevos-limites-velocidad-2022>, accedido el 22 de diciembre de 2022.
- [44] C. Brunner, A. Garavaglia, M. Mittal, M. Narang, and J. V. Bautista, "Inter-system handover parameter optimization," Disponible en: https://www.researchgate.net/publication/224761739_Inter-System_Handover_Parameter_Optimization, 2006, accedido el 22 de diciembre de 2022.
- [45] aws, "Que es la vision artificial," Disponible en: <https://aws.amazon.com/es/what-is/computer-vision/>, s/f, consultado el 26 de enero de 2022.
- [46] J. Wexler, *Get Programming with Node.js*. Manning, 2019.
- [47] NVIDIA. Jetpack sdk 4.6.3. Accedido el 17 de junio de 2023. [Online]. Available: <https://developer.nvidia.com/jetpack-sdk-463>
- [48] ——. Jetpack sdk 4.6.3. Accedido el 17 de junio de 2023. [Online]. Available: <https://developer.nvidia.com/jetpack-sdk-463>
- [49] Kangalow. Jtop: The ultimate tool for monitoring nvidia jetson devices - jetsonhacks. Accedido el 17 de junio de 2023. [Online]. Available: <https://jetsonhacks.com/2023/02/07/jtop-the-ultimate-tool-for-monitoring-nvidia-jetson-devices/>
- [50] N. Escobar. (2015) Para que sirve el swap en linux y como cambiarlo. Accedido el 19 de junio de 2023. [Online]. Available: <https://hipertextual.com/2015/09/swap-en-linux>
- [51] Nakai-Omer. Github - mdegans/nano_build_opencv: Build opencv on nvidia jetson nano. Accedido el 20 de junio de 2023. [Online]. Available: https://github.com/mdegans/nano_build_opencv
- [52] W. Electronics. Sim7600g-h 4g for jetson nano - waveshare wiki. Accedido el 16 de abril de 2023. [Online]. Available: https://www.waveshare.com/wiki/SIM7600G-H_4G_for_Jetson_Nano_4G_connecting
- [53] Amazon. Amazon amplify. Accedido el 25 de diciembre de 2022. [Online]. Available: <https://aws.amazon.com/es/amplify/>