ITFUSION SAS

Documento Funcional

(High Level Design)

Plataforma tecnológica para transporte terrestre que disminuye accidentes automovilísticos

Ver: 1.1

28 de febrero, 2023

Importante Aviso de Confidencialidad

La información contenida en todas las paginas de este documento constituye información confidencial de Itfusion SAS Colombia.

Este documento incluye información respecto del proyecto presentado y avalado por SENA/MINCIENCIAS en la convocatoria

Información del documento

Título del documento
Arquitectura de la Solución para el Proyecto SENA/MINCIENCIAS…

Preparado por

Daniel Ricardo Bernal

Última revisión

28 de febrero, 2023

Fecha de creación del documento

17 de febrero, 2023

Versión del documento

2.0

Fecha de la versión

16 de diciembre, 2023

Contents

Introducción

1.1 Propósito del documento

El propósito de este documento es definir la arquitectura funcional y física para el proyecto PTTTAA (Plataforma tecnológica para transporte terrestre que permite disminuir accidentes automovilísticos)

Este documento contiene:

- - La arquitectura general y funcional
- - El despliegue de la solución en todos sus componentes y sus características a nivel de recursos tecnológicos.
- - La arquitectura física
- - Las características y modelo de las bases de datos requeridas.
- - Descripción de dispositivos (Diadema)
- - El plan de implementación de alto nivel.

1.2 Documentos relacionados

- Propuesta_FormularioProyectos93615

Alcance del proyecto

El alcance del proyecto se engloba de los siguientes objetivos:

Objetivo General:

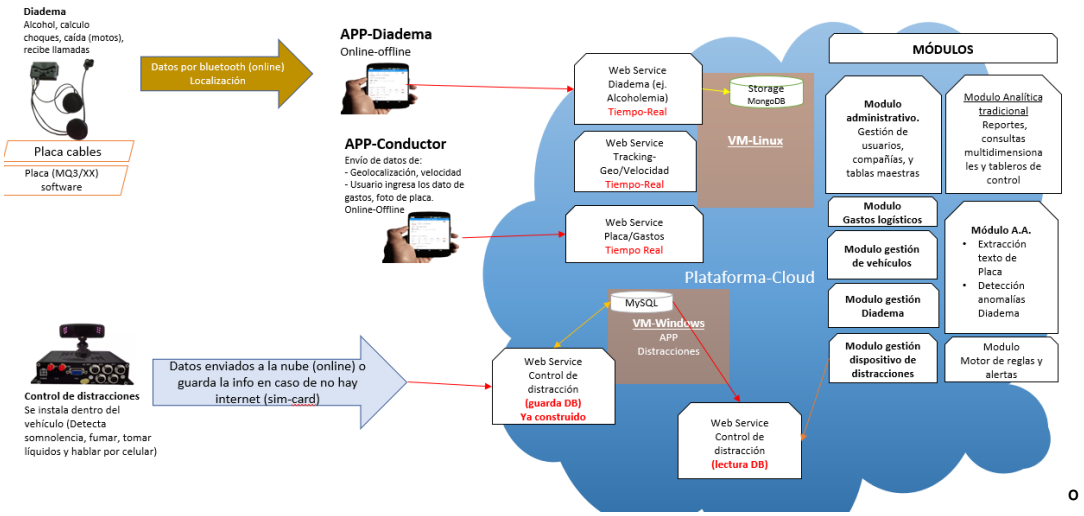
Desarrollar una plataforma integrada con internet de las cosas, Inteligencia Artificial y analítica de datos, que permita conectar los vehículos y conductores, para establecer parámetros de seguridad vial, controlar costos, ayudar a proteger el medio ambiente, medir la gestión y determinar mejoras en el servicio prestado.

Objetivos específicos:

- Realizar la definición y adquisición de la infraestructura requerida, así como la construcción de los diseños, arquitectura y la planeación detallada para la implementación de la solución. El despliegue se hará sobre nuevas máquinas virtuales (VMs).
- Desarrollar web services, integrar y realizar pruebas unitarias con dispositivos de detección de distracciones, cámaras de monitoreo y dispositivos de detección de alcoholemia.
- Realizar la adecuación de la nueva versión del dispositivo de detección de alcoholemia o “diadema inteligente”, así como los ajustes requeridos en la app móvil y en la interfaz de administración de la plataforma central.
- Desarrollar módulo de analítica tradicional como analítica avanzada tanto en tiempo real como de históricos. Esto incluye la creación de un módulo de alarmas pre construidas que permitirá ser la base para en una siguiente fase y luego del proyecto construir un motor de reglas y alarmas de manera parametrizable.
- Realizar una validación pre-comercial integral de la solución implementada con clientes potenciales y proceder con los ajustes requeridos, a fin de medir y analizar el desempeño en ambientes reales.
- Realizar la transferencia tecnológica al SENA.

Ambiente de Producción

3.1 Arquitectura general de la solución



Arquitectura general de la solución

La arquitectura definida para este proyecto tiene por objetivo desarrollar las siguientes funcionalidades:

1. **Dispositivo Diadema:** Dispositivo portátil que va en la cabeza del conductor que permite leer el tiempo real las emisiones de los gases de la boca del conductor y convertirlos en grados de alcohol como son clasificados en Colombia para efectos de conducción. Para este componente de detección de alcohol, se realizarán pruebas controladas como de campo, con hardware que permita detectar si el conductor se encuentra apto o no para conducir según las normas legales**.**Adicionalmente, este dispositivo podrá calcular un posible choque frontal de un carro, más la realización de pruebas con vehículos de control remoto que basados en fuerzas G puedan determinar por medio de software si hubo un posible choque. También el dispositivo podrá detectar posibles caídas laterales para casos de los motocicletas, recibir llamadas y enviar esta información en tiempo real por medio de bluetooth al Smartphone. Para efectos del proyecto solo se afinará la versión diadema de motos ya que se podrán alcanzar mejores resultados en la detección de alcoholemia basado en los gases de la boca de la persona dentro del casco del motociclista. Para versión diadema de carros se realizarán diferentes pruebas y recolección de datos que sirva para refinar en una siguiente fase posterior a este proyecto donde existan muchos más datos en diferentes ambientes.

La Diadema físicamente contiene dos placas: una placa pequeña para transducir los gases de la persona y convertirlos en grados de alcoholemia. Estos grados de alcoholemia son enviados a la otra placa (más grande) el cual procesa la información que serán transmitidos al dispositivo móvil a través de bluetooth.

A continuación, se adjunta una imagen con el diagrama de bloques de la diadema.

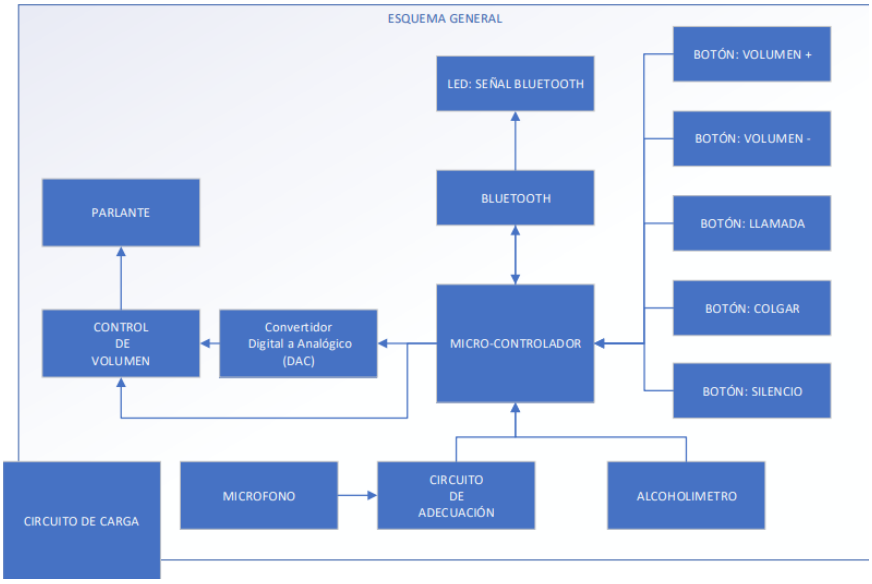


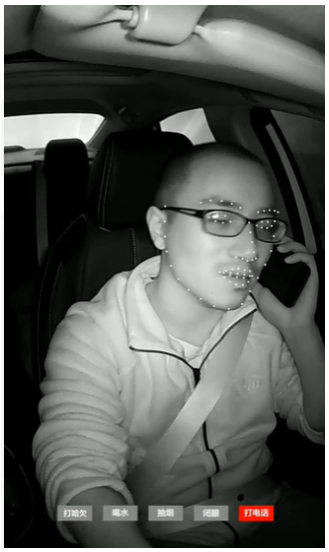
Diagrama de bloques general de la diadema

2. **Aplicación (APP) Diadema**: Esta aplicación toma los datos que envía el dispositivo diadema, y junto con los datos de velocidad y localización son enviados en tiempo real a la plataforma web de Itfusion en caso de tener conexión a internet. Si no tiene conexión a internet, los datos son guardados en el teléfono inteligente hasta cuando recobre el internet, y en este caso los datos son sincronizados a la plataforma.

3. **Aplicación (APP) Conductor**: Esta app permite capturar de manera automática los datos de localización por donde va cada conductor, con su velocidad y son enviados a la plataforma si existe conexión a internet. Si no existe conexión a internet, los datos son guardados en la app móvil localmente y luego al recobrar el internet, los datos son sincronizados a la plataforma. Además de esto, de manera manual el conductor puede tomar los datos de gastos que tiene en cada viaje, como gasolina, peajes, y un valor de otro general con su descripción, y con toma de foto de la placa para ser cargados en tiempo real a la plataforma.

4. **Dispositivo de distracciones:** Es un dispositivo que se instala dentro del vehículo para detectar distracciones como somnolencia, hablar por celular, fumar, tomar líquidos y en caso de detectar la distracción, suena una alarma interna dentro del vehículo. Además de esto, este dispositivo envía en tiempo real los datos de estas alertas a la plataforma de Itfusion en caso de haber internet. Si no hay internet esta información es guardada en el dispositivo tipo MDVR local del vehículo y

luego al recobrar la conexión de internet, los datos son sincronizados a la plataforma.



5. **Módulos de la solución:**

- **Módulo administrativo**, administración del sistema, desde el “super usuario” permitir administración de compañías, cantidad de usuarios por compañía y fecha de vencimiento, además gestionar las tablas maestras del sistema como tipos de vehículos (modelos y marcas), tipo operación (Bus urbano, bus regional, carga, infraestructura, automóvil), tipo de vehículo (2 ejes, 3 ejes, articulado). Desde un usuario administrativo de la compañía, crear, actualizar y desactivar usuarios (Solo tendrá dos tipos o roles de usuario por compañía, 1 el administrador de la compañía que ver y actualizar todo lo de la compañía y 2 el usuario que solo puede consultar o ver reportes y tableros de control, además de crear, actualizar o asignar vehículos, diademas o dispositivos de distracciones).

- **Módulo gastos logísticos**: En cada ruta o servicio prestado por un vehículo, el conductor desde la app móvil registrará los gastos logísticos como peajes, kilometraje inicial y final, combustible inicial y final con la foto de la placa del vehículo, y será integrado por un webservice en la plataforma. Adicionalmente se llevará un reporte de gastos logísticos en la plataforma web.

- **Módulo de gestión de vehículos**. Parametrizar datos como kilometraje inicial, peso, tipo por cada vehículo. También los tipos de vehículos se cargarán por archivo en Excel recolectada directamente a la base de datos o se podrán agregar nuevos desde la interfaz web. Se podrá asignar un vehículo a un usuario conductor.

- **Módulo de gestión diadema.** En este módulo permitirá visualizar en tiempo real 1 tablero de conductores en estado de alcoholemia, 1 reporte de historial de datos de alcoholemia. Para integrar la información de la diadema, se creará un webservice en la plataforma que permitirá guardar los datos de este dispositivo. Se podrá asignar una diadema a un conductor.

- **Módulo de gestión dispositivo de distracciones**. A través de un webservice se consumirán los datos generados por el dispositivo de distracciones consumiendo la base de datos MySQL que almacena el sistema CMSV6 que describimos más abajo y estos serán visualizados en 1 tablero de control, y 1 reporte dentro de la plataforma IoT de Itfusion. Se instalará para este proyecto un dispositivo de distracciones a un vehículo con 1 cámara.

  Sistema CMSV6: Esta plataforma se instala localmente en windows, permite integrar el dispositivo de distracciones para realizar procesamiento de vídeo, con transmisión de archivos multimedia desde el MDVR para realizar el seguimiento efectivo de vídeo en tiempo real, detectar la ubicación por GPS, almacenar archivos de vídeo, detectar fatiga, malos hábitos de conducción y otros. Esta información es almacenada tanto en directorios del servidor con en el motor de bases de datos MySQL. Finalmente se accederá esta base de datos para ser consumidos por Webservices de la plataforma de Itfusion y que permitirán mostrar reportes y consultas desde la misma.

- **Módulo motor de reglas y alarmas**. Para efectos de este proyecto se crearán 2 alarmas con fórmulas preconstruidas a nivel de programación con mensaje al correo electrónico, sin embargo, se dejará una tabla en la base de datos que almacenará cada fórmula de tal manera que en futuros desarrollos el usuario final pueda crear de manera parametrizable cualquier alarma configurando cada fórmula sin conocimiento técnico, uno de los más altos valores de una plataforma IoT que requiere alto dinamismo y agilidad.

- **Módulo de analítica tradicional.** Se crearán los siguientes reportes de históricos, y tableros de control en tiempo real y de histórico, y consultas dinámicas multidimensional autoservicio:
  - 1 tablero de control en tiempo real de datos de alcoholemia de los conductores.
  - 1 reporte del historial de información proveniente de la app de la diadema
  - 1 tablero de control de historial del módulo de distracciones
  - 1 reporte de información de distracciones
  - 1 reporte de la información de gastos logísticos
  - 1 consulta multidimensional autoservicio de información histórica de la app de conductores con datos de velocidad, localización, nivel de alcohol, si choco, o si cayo en caso de un motociclista.

- **Módulo de Analítica Avanzada:**
  a. **Extracción del texto de placa desde una foto**. Con el uso de la Inteligencia artificial permite a través de un webservice extraer el texto del número de placa del conductor desde una foto para ser almacenada en la base de datos.
  b. **Detección de anomalías**. Con el uso del Machine Learning, permite clasificar automáticamente si un valor de alcoholemia proveniente de la diadema es anómalo o no y guardarlo en base de datos clasificado. Para poder realizar esto se tomará una muestra proveniente de la diadema y en el momento de la las pruebas, también tomar en el mismo momento una prueba usando el dispositivo real y profesional de alcoholemia, se guardarán y compararán ambos valores para realizar la clasificación y

detección de anomalías en los datos de alcoholemia en los conductores. Este proceso inicialmente se ejecutará en batch para el proyecto, pero se deja diseño arquitectónico para implementar a futuro a corto plazo para que el proceso sea ejecutado en tiempo real.

6. **Web Services**

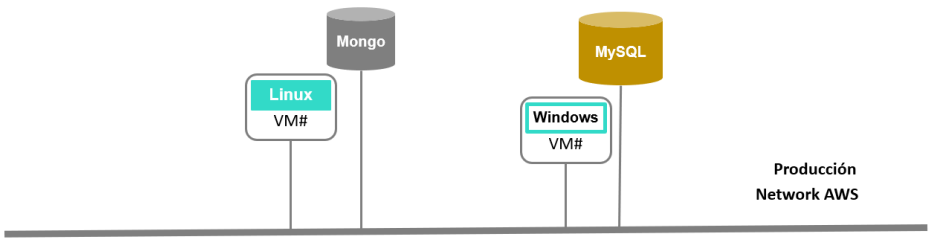
- **Diadema:** Recibe datos de la "APP Diadema", solo y cuando detecte estado de alcoholemia en el dispositivo físico, estaría recibiendo estos datos en la plataforma, junto a la localización, fecha en que ocurrió el evento, completando datos en caso de haber ocurrido un choque, o una caída (motos) y velocidad del vehículo. Estos datos son almacenados en la base de datos, siendo analizados en tiempo real en los tableros de control, y por el Módulo de motor de reglas.

- **Tracking:** Recibe datos de la "APP Conductor" cada vez, cumpla con la distancia definida en el sistema, por compañía y conductor, como la localización, fecha del usuario/conductor, y velocidad por cada trayecto (viaje). Estos datos son almacenados en la base de datos, siendo analizados en tiempo real en los tableros de control, y por el Módulo de motor de reglas.

- **Placa/Gastos:** Recibe datos de la "APP Conductor" por cada trayecto como gasolina, peajes, y un valor de otro general con su descripción, y con toma de foto de la placa para ser cargados en tiempo real a la plataforma, luego de estos son almacenados en la base de datos, y la imagen de la placa son almacenadas en un directorio en la plataforma, como también extrae el texto de la foto de la placa que será almacenado dentro de la DB.

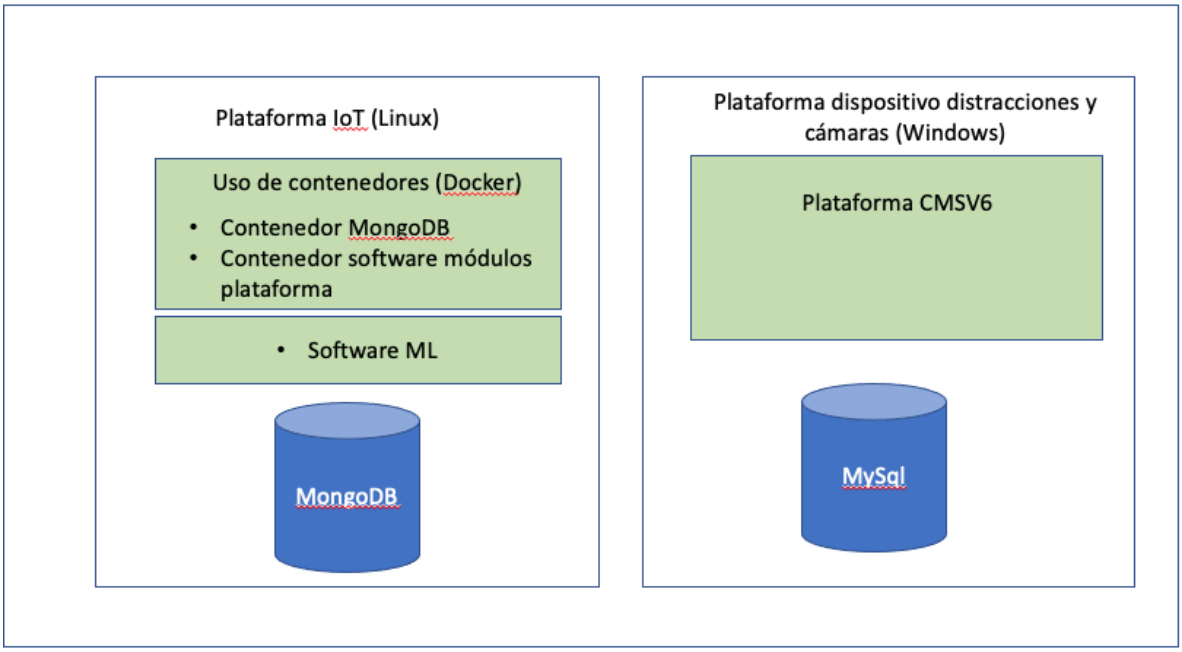
- **Control distracciones:** lee información de la base de datos MySQL donde el sistema CMSV6 almacena los datos para ser consumido a través de un WS y ser mostrado en reporte y tablero de la plataforma.

3.2 Arquitectura de red



Arquitectura de red en producción

1. Servidores:
   Arquitectura de servidores en producción:



Arquitectura de servidores en producción

a. Servidores linux

La arquitectura tecnológica de la plataforma de este servidor será sobre contenedores con la herramienta docker, la cual tiene múltiples beneficios como escalamiento, balanceo de cargas, modularidad, facilidad de instalación del stack tecnológico de aplicaciones y software, restauración, implementación rápida, ahorro en costos por el uso de menos hardware, sistemas de actualizaciones constantes orquestada, y otras.

Proveedor: Amazon AWS, con 1 servidor para producción y un 1 servidor para desarrollo y pruebas.

Requerimientos del servidor:

- Sistema operativo Linux: Ubuntu 20.04
- RAM: 16 GB
- CPU: 4 CPUs
- Disco Duro: 100GB SSD

Especificaciones adicionales:

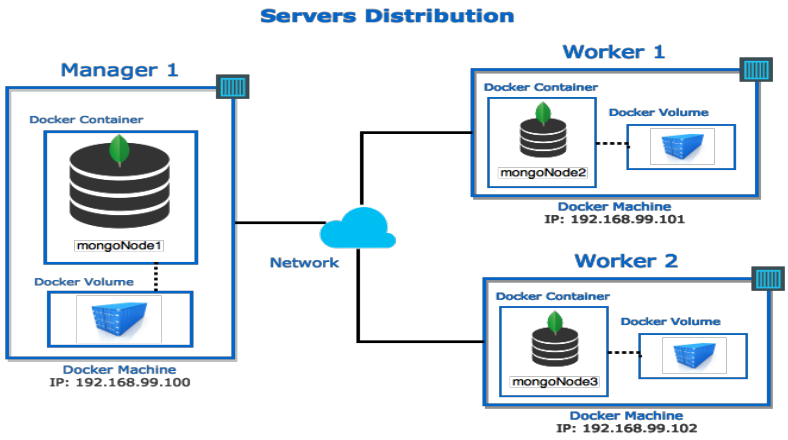
i. NodeJS v16.13.0 o superior. Se instalará sobre un mismo contenedor de docker junto con npm. Se instalará sobre un mismo contenedor de docker junto con NodeJS
ii. Npm 8.19.1 o superior.
iii. MongoDB 6.0.

Beneficios de usar MongoDB

1. Este motor de base de datos nos permite tener flexibilidad de adaptar su diseño de colecciones, con diferentes tipos de objetos y campos, cambiarlos en el

tiempo y poder agregar diferentes tipos de dispositivos de la misma manera para convertirla en una plataforma IoT.

2. MongoDB es altamente reconocida por su alto desempeño y uso en aplicaciones con analítica en tiempo real para grandes volúmenes de datos no estructurados.
3. MongoDB permite escalado horizontal automático, clustering, y replicación automática “sharding”.
4. MongoDB permite manejo de series temporales de manera nativa para la versión 6.0, lo cual nos podrá mejorar ampliamente el desempeño al integrar miles de dispositivos conectados que generan información por segundos.
5. Como lo muestra la siguiente figura, nuestra arquitectura esta diseñada para soportar múltiples instancias, que nos permitirá a futuro procesar o analizar miles de millones de datos de manera eficiente, escalable y con datos provenientes de los dispositivos conectados a la plataforma en tiempo real, permitiendo tener una alta tolerancia a fallos, alta escalabilidad, disponibilidad y seguridad. Esta arquitectura estará soportada por contenedores sobre docker, y en esta primera fase de este proyecto solo se instalará una instancia de MongoDB, pero luego será fácilmente escalada a múltiples instancias en la medida que se vaya requiriendo o hasta pasar a MongoDB Atlas.



Seguridad: Para garantizar la seguridad de mongoDB, un control de acceso al motor basado en roles, con contraseñas de alta seguridad.

Backups: Los backups de la base de datos MongoDB se realizarán diariamente en este proyecto y se construirá un cron que permita ejecutar esta tarea de manera automática y

almacenada en un disco interno como también será enviada por correo en formato zip.

iv. Docker
v. PHP 7.3 o superior.
vi. Python. Se instalará junto con OpenCV y Tensorflow en el mismo ambiente.
vii. OpenCV.
viii. Tensorflow.
ix. Apache web server con SSL. Se conectará con el dominio protégeme.com.co.

b. Servidor Windows:

Proveedor: Amazon AWS:1 servidor.

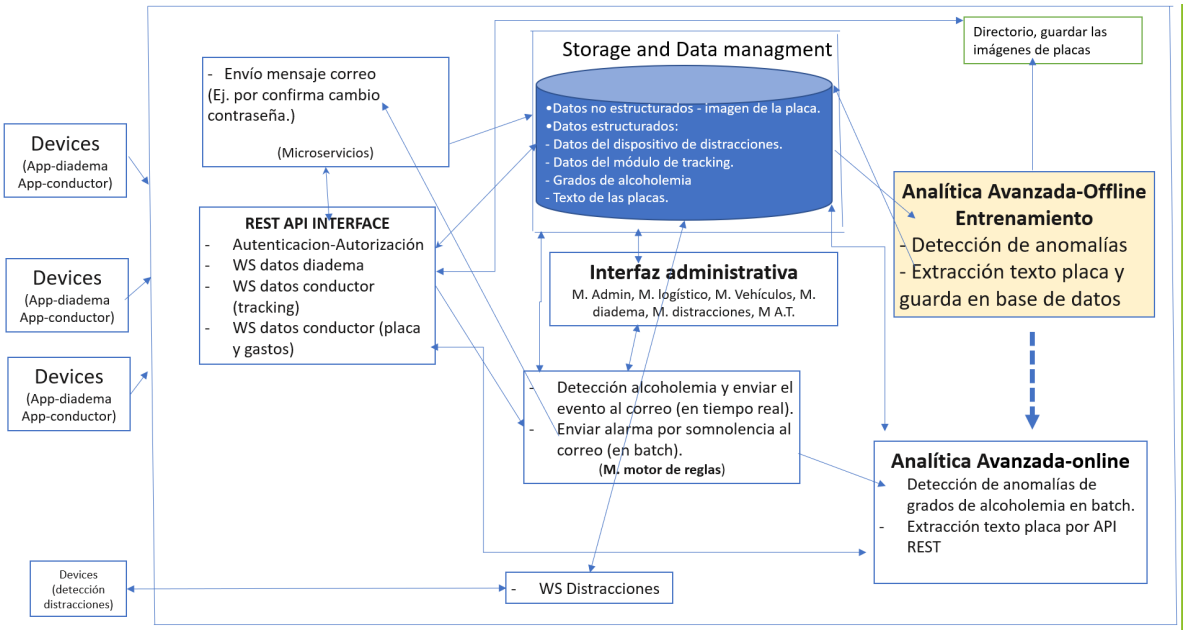
Requerimientos del servidor:
- Sistema operativo Microsoft Windows Server 2016 (sistema operativo de 64 bits)
- RAM: **8** GB o superior
- CPU: 4 CPUs
- Disco Duro: 100GB SSD

2. Instalar instancias:
Linux Ubuntu:
• Inicializar nuestra instancia con Ubuntu 20.04 en la nube
• Instalar pilas de software básico y avanzado, según sea necesario
• Agregar perfiles de seguridad a la pila fundamental
• Crear y configurar una base de datos de series temporales
• Dar a nuestra plataforma un nombre de dominio

Windows:
• Inicializar nuestra instancia en Windows Server 2016 en la nube
• Instalar plataforma CMSV6
• Agregar perfiles de seguridad a la pila fundamental
• Dar a nuestra plataforma un nombre de dominio

3.2 Despliegue de componentes

En la siguiente figura se muestran los componentes que harán parte de la solución:



Arquitectura funcional de la solución

Device: El proyecto contará para este proyecto con 3 tipos de dispositivos: Diadema, Detección de distracciones y Smartphones.

Microservicios: Modulo funcional que será utilizado en diferentes partes de la solución como:

- Envió de mensaje por correo electrónico: En el momento de generar una alarma por somnolencia.
- Envió de mensaje por correo electrónico: En algún cambio de contraseña.

Rest API: La solución manejará 4 tipos de webservice, necesarios para la publicación (pub) y consulta de información. Cada uno de estos WS maneja su propia autenticación:

- Web Service – diadema
- Web Service – datos del conductor (tracking)
- Web Service – datos del conductor (placa y gastos)
- Web Service – distracciones.

Storage / Data Management: Almacenamiento de los siguientes tipos de datos

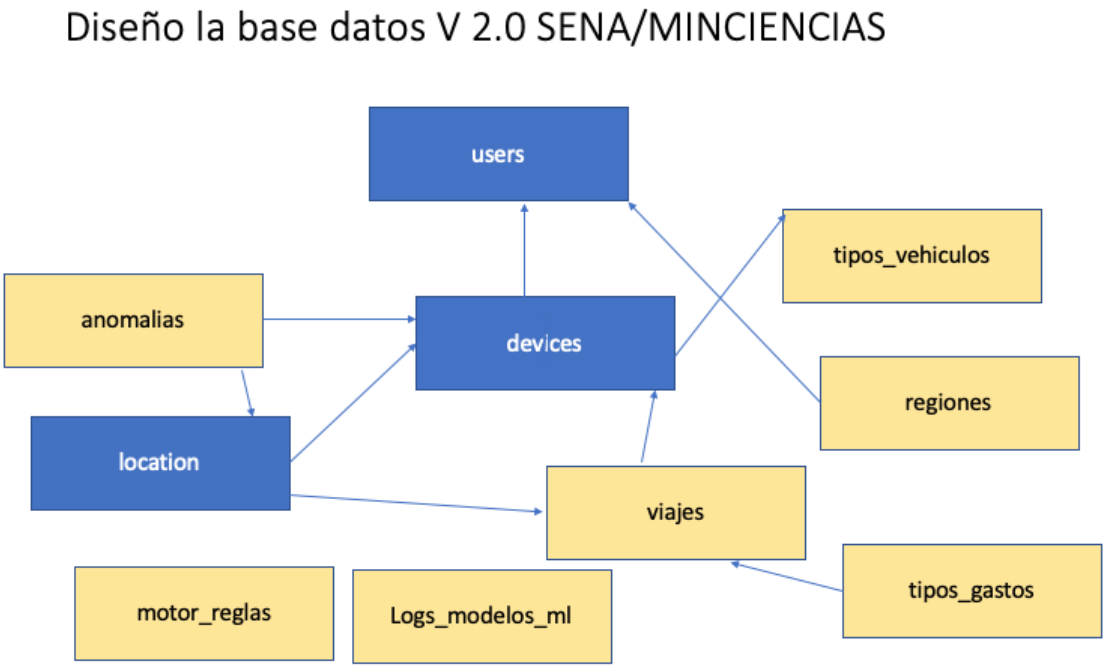
- Datos no estructurados como la imagen de la placa.

- Datos estructurados:

- Datos generados por el dispositivo de distracciones de conductores.
- Datos enviados por el módulo de tracking.
- Grados de alcoholemia como el Identificador del device del conductor, fecha, estado de alcoholo (SI/NO), ubicación, kilometraje, accidente 0/1.

- Texto de las placas.

La base de datos usados para almacenar la información de la plataforma estará sobre Mongo versión 6.0 o mayor. Tendrá el siguiente diseño donde en color azul se muestran las colecciones de la plataforma de tracking actual de Itfusion SAS, y las de color amarillo son tablas que se estarían implementando en este proyecto.



Diseño Base de datos de la plataforma

Los siguientes son los campos de cada una de estas colecciones nuevas y antiguas:

1. Nuevas colecciones de la base de datos “maps”
    a. tipos_vehiculos
        i. _id (obligatorio y automático por la base de datos)
        ii. tipo (motocicleta, vehículo pequeño, carga liviana, carga pesada, etc). (obligatorio)
        iii. clase. (obligatorio)
        iv. marca. (obligatorio)
        v. linea (obligatorio)
        vi. peso (obligatorio)
        vii. clindraje (obligatorio)
        viii. activo (obligatorio)
        ix. created_time (obligatorio)
    b. tipos_gastos
        i. _id (obligatorio y automático por el la base de datos)
        ii. nombre
        iii. activo (obligatorio)
        iv. created_time (obligatorio y automático por la base de datos)
    c. motor_reglas

i. _id (obligatorio y automático por la base de datos)
ii. Nombre (obligatorio)
iii. Activa (obligatorio)
iv. topico_patron (obligatorio)
v. payload_patron (obligatorio)
vi. método (obligatorio)
vii. webhook (obligatorio)
viii. userId (Compañía a la que aplican las reglas) (obligatorio)

d. anomalias (guarda cada anomalía registrada en tiempo real o en proceso en batch con la detección de anomalías del módulo A.A. para posterior reporte de anomalías)
   i. _id (obligatorio y automático por la base de datos)
   ii. deviceId (obligatorio)
   iii. anomalia (0=NO, 1=SI) (obligatorio)
   iv. created_time (obligatorio)
   v. id_location (obligatorio)
e. logs_modelos_ml (Modelos ML aplicados)
   i. nombre (nombre del modelo) (obligatorio)
   ii. porcentaje_efectividad (obligatorio)
   iii. activo (obligatorio)
   iv. crated_time (obligatorio)
   v. fecha_desactivacion (obligatorio)
f. regiones
   i. _id (obligatorio y automático por la base de datos)
   ii. codigo_ciudad (obligatorio)
   iii. nombre_ciudad (obligatorio)
   iv. codigo_departamento (obligatorio)
   v. nombre_departamento (obligatorio)
   vi. codigo_pais (obligatorio)
   vii. nombre_pais (obligatorio)
   viii. created_time (obligatorio)

g. viajes. Por cada vehículo se registrará un consecutivo de viaje.
   i. _id (obligatorio y automático por la base de datos)
   ii. deviceId (obligatorio)
   iii. no_viaje **(**consecutivo numérico por viaje por dispositivo). (obligatorio)
   iv. cantidad_frenadas_bruscas. (obligatorio)
   v. cantidad_aceleradas_bruscas (obligatorio)
   vi. municipio_inicial (nombre del municipio donde inicio el viaje) (obligatorio)
   vii. municipio_final (nombre del municipio donde finalizo el viaje) (obligatorio)
   viii. fecha_hora_inicio (fecha y hora del inicio del viaje) (obligatorio)
   ix. fecha_hora_fin (fecha y hora del fin del viaje) (obligatorio)
   x. promedio de velocidad (obligatorio)
   xi. estado (Finalizado sin gastos:1, Finalizado con gastos:2) (obligatorio)
   xii. created_time (obligatorio)

xiii. gastos. Son los gastos que se registran al finalizar cada viaje y cada tipo de gasto pertenece a uno de los registros de la colección tipos_gastos

2. Nuevos campos de colecciones existentes
    a. location
        i. aceleración (ya incluido y probado luego de inicio del proyecto). (obligatorio)
        ii. alcohol. (obligatorio)
        iii. concentración (obligatorio)
        iv. colision (obligatorio)
        v. caida (obligatorio)
        vi. recorded_at (obligatorio, es la fecha recolectada del Smartphone que reporta datos). (obligatorio)
        vii. id_viaje (sale de la colección viajes) (obligatorio)
    b. users
        i. expiration_date (obligatorio)
        ii. seguimiento_metros (para rol: Compañía si aplica para toda la compañía o si es usuario normal, se aplica solo para ese usuario el valor definido). (obligatorio)
        iii. emails (correos a notificar en caso de anomalía de alcoholemia en caso de la diadema, o somnolencia en caso de dispositivo de distracciones) (obligatorio)
        iv. id_pais (57 por defecto para usuario admin compañia). (obligatorio)
        v. id_region (obligatorio)
        vi. tipo_viajes (manuales, automáticos). Por defecto manuales, requieren ingresar kilometraje y nivel de gasolina inicial al comenzar el viaje a través de un botón en la app, y requiere finalizar el viaje a través de un botón registrando kilometraje y nivel de gasolina final. (obligatorio si es viaje tipo manual)
        vii. no_minutos_viaje (cuantos minutos luego estar parado se considera nuevo viaje. Por defecto 5 si es automático). (obligatorio)
        viii. gastos_obligatorio (0=No, 1=SI por viaje) (obligatorio)
        ix. id_viaje_actual (valor numérico auto incremental por viaje del usuario). (obligatorio)
        x. velocidad_maxima_ciudad_privado. Obligatorio por tipo de usuario compañia
        xi. velocidad_maxima_carril_simple_nacionales_privado. Obligatorio por tipo de usuario compañia
        xii. velocidad_maxima_doble_carril_nacionales_privado. Obligatorio por tipo de usuario compañia
        xiii. velocidad_maxima_ciudad_publico. Obligatorio por tipo de usuario compañia
        xiv. velocidad_maxima_carril_simple_nacionales_publico. Obligatorio por tipo de usuario compañia
        xv. velocidad_maxima_doble_carril_nacionales_publico. Obligatorio por tipo de usuario compañia

    c. devices

i. type:Diadema, dispositivo distracciones, Smartphone, botón de pánico, GPS vehículo, GPS bicicleta, vehículo motorizado. (obligatorio)
ii. no_placa (Solo ingresar si es vehículo motorizado). (obligatorio)
iii. modelo (año) (obligatorio si tiene relación con la colección tipos_vehiculo)
iv. kilometraje_inicial (obligatorio si tiene relación con la colección tipos_vehiculo)
v. kilometraje_actual (obligatorio si tiene relación con la colección tipos_vehiculo)
vi. tipo_motor (gasolina, diésel, eléctrico, híbrido gasolina-eléctrico, híbrido diésel-elécctrico) (obligatorio si tiene relación con la colección tipos_vehiculo)
vii. id_tipo_vehiculo. (obligatorio si tiene relación con la colección tipos_vehiculo)
viii. tipo_servicio. Lista (Público, Privado). (obligatorio si tiene relación con la colección tipos_vehiculo)
ix. odómetro (por defecto 0 para iniciar, pero se podría actualizar en la plataforma)
x. seguimiento_metros (por defecto 10 metros). (obligatorio si tiene relación con la colección tipos_vehiculo)
xi. serial:si es tipo de dispositivo de distracciones, registrar el No de serial

La base de datos para los datos relacionados a los dispositivos de detección de distracciones será MySQL.

Rule Engine (motor de reglas): El sistema generará dos tipos de eventos de alarmas (notificación).

- Una notificación de alcoholemia será creada al tomar la información en tiempo real del WS de la diadema. Posterior se guardará en base de datos, y se enviaría un correo electrónico al administrador de la solución.

  Identificador del usuario/conductor (nombre, documento identidad)
  Fecha que ocurrió el evento
  Estado de alcoholemia (0 = NO o 1 = SI)
  Velocidad
  Localización

- Una notificación de distracción (somnolencia) será creada al tomar la información de la base de datos. Posterior se guardará en base de datos, y se enviaría un correo electrónico al administrador de la solución.

  Identificador del usuario/conductor (nombre, documento identidad)
  Fecha que ocurrió el evento
  Tipo de distracción: somnolencia

Velocidad
Localización

Application and User Management

- Administración de la plataforma - usuarios por compañía (multicompañía)
- Tablero en tiempo real de conductores con alcoholemia
- Tablero en tiempo real conductores exceso de velocidad

Módulo de Analítica Avanzada

Se implementarán dos modelos de Machine Learning (ML), clasificados en las siguientes fases

- Entrenamiento (offline):

  Detección de anomalías, basados en los datos de alcoholemia de la diadema que se encuentran en la base de datos, se adecua la información para construir el modelo de ML, y así optimizarlo para aplicarlo en producción.

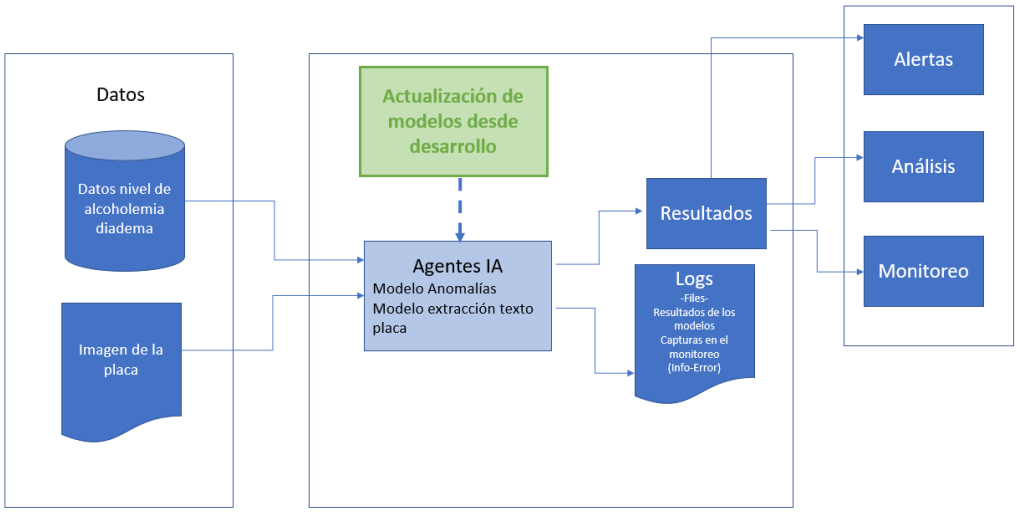
  Extracción del texto de la placa y almacenamiento en DB: Basado en las fotos recolectadas y almacenadas en un directorio, se construye el modelo de Visión Artificial, que permite leer y extraer el texto de cada una de las placas de los vehículos.

- Online:

  Detección de anomalías: permite cada 24 horas corre el proceso en batch de ML (detección de anomalías) para clasificar la información, y actualizando en la base de datos como anómala o no anómala, y pendiente por clasificar.

  Extracción del texto de la placa por API Rest: Basado en foto recolectada del WS, y almacenadas en un directorio, se aplica el modelo de Visión Artificial, que permite leer y extraer el texto de cada una de las placas de los vehículos, para luego ser almacenada en la base de datos.

En la siguiente gráfica se describe como se realiza el proceso de consumo de información desde los modelos de Machine Learning (componente Agentes IA) en producción, desde donde son actualizados los mismos por el componente “Actualización de modelos desde desarrollo” que se encuentra en color verde, y finalmente generan unos resultados para ser consumidos por sistemas de alertas, analítica o monitoreo, y logs para registrar el procesamiento de los modelos.



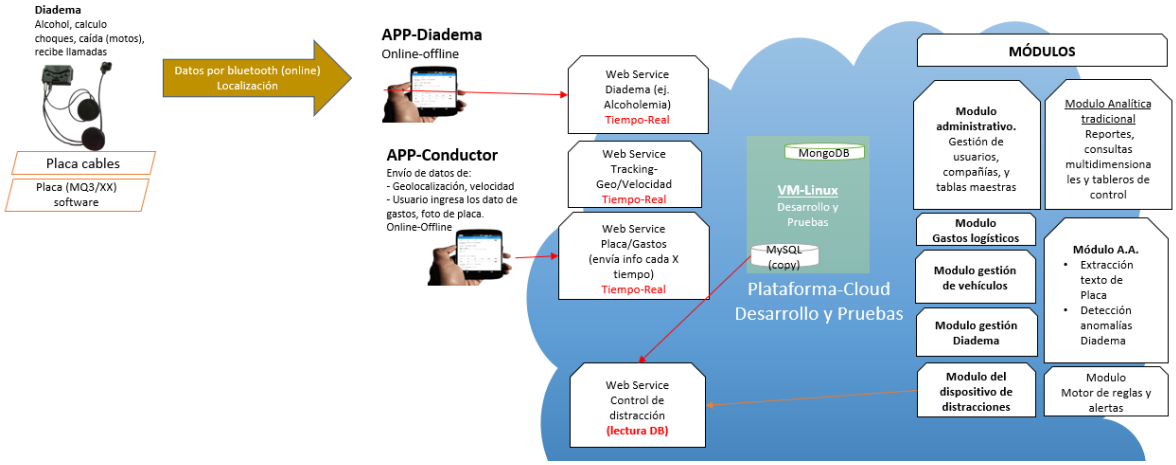
Ciclo de vida ML: Alto nivel

El anterior gráfico se observa los datos de entrada correspondientes al estado de alcoholemia que llega de la diadema, y la imagen de la placa extraída por el componente API-REST, para luego ser procesados por los Modelos de ML definidos dentro del módulo de analítica avanzada.

Los componentes **Resultados, Logs, Alertas, Análisis y Monitoreo** recibe la información generada por los modelos de Detección de Anomalías o Extracción de Texto de la Placa de manera automática, generando Resultados que serán cargados por los componentes de Alertas, Análisis y/o Monitoreo. A través de los Logs, los modelos, se estarán revisando continuamente para validar si es necesario o no cambiarlos por un modelo más eficiente.

Ambiente de Desarrollo

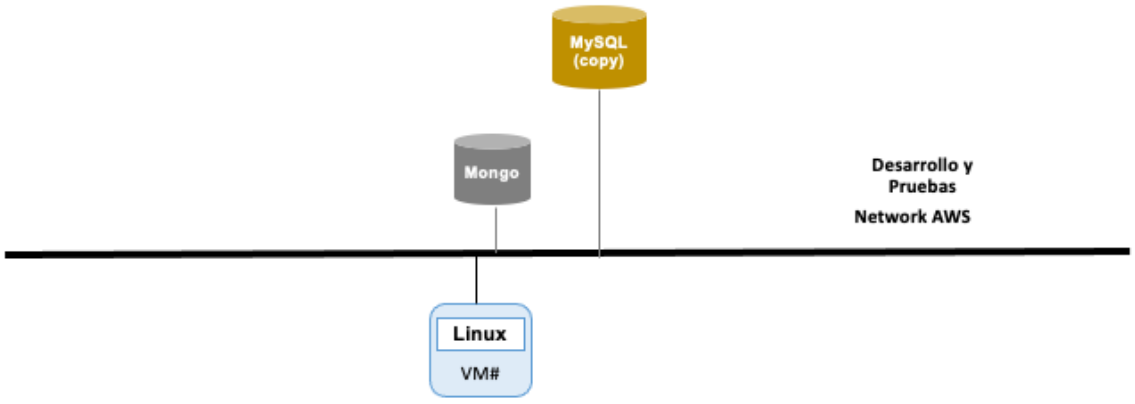
4.1 Arquitectura general de la solución



Arquitectura general de la solución en desarrollo

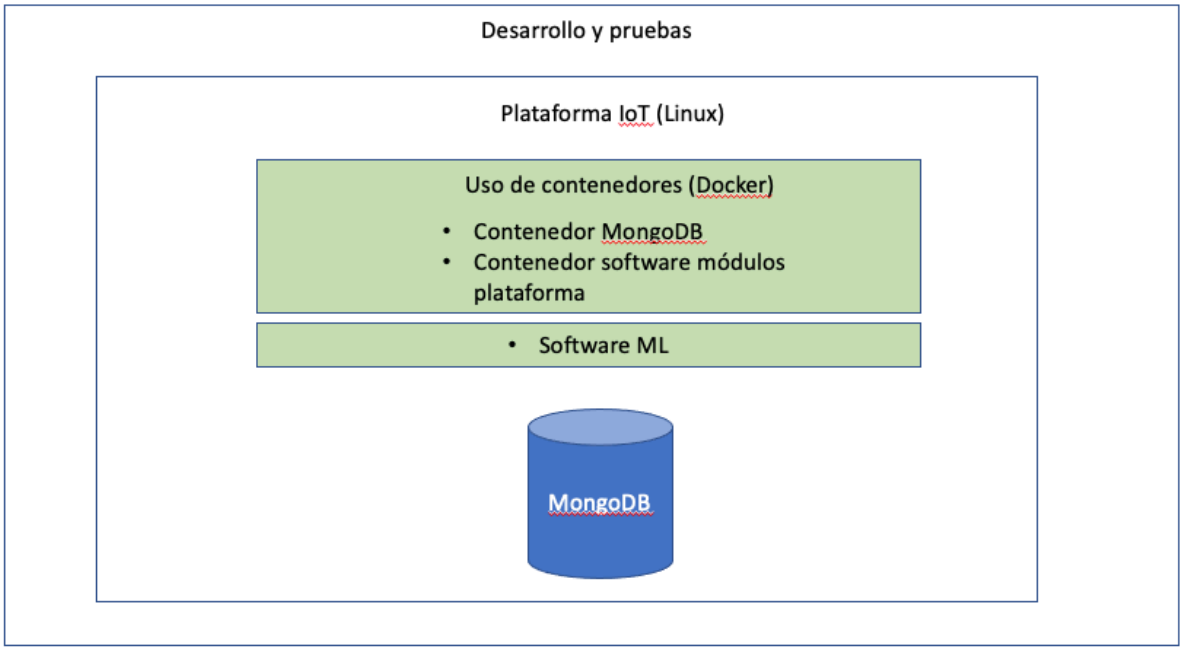
A diferencia del ambiente de producción, en desarrollo no estaría contemplado el componente de “control de distracciones”, solamente se utilizará el WebService para hacer consultas hacia la base de datos MySQL.

4.2 Arquitectura de red



Arquitectura de red en desarrollo

A continuación, se describe el servidor de desarrollo y pruebas:



Arquitectura de servidores en desarrollo y pruebas

a. La arquitectura tecnológica de la plataforma de este servidor será sobre contenedores con la herramienta docker, la cual tiene múltiples beneficios como escalamiento, balanceo de cargas, modularidad, facilidad de instalación del stack tecnológico de aplicaciones y software, restauración, implementación rápida, ahorro en costos por el uso de menos hardware, sistemas de actualizaciones constantes orquestada, y otras.

   Proveedor: Amazon AWS, con 1 servidor para producción y un 1 servidor para desarrollo y pruebas.

   Requerimientos del servidor:
   - Sistema operativo Linux: Ubuntu 20.04
   - RAM: 4 GB o superior.
     - NOTA: Para efectos de fase de entrenamiento de los 2 componentes de ML del proyecto, se requerirá subir a 32 GB de RAM este servidor por máximo 2 meses, con 16 CPUs y disco SSD.
   - CPU: Intel Xeon Quad-core (4 núcleos) (2.33GHz)
   - Disco Duro: 100GB SSD

   Especificaciones adicionales:
   i. Docker
   ii. NodeJS v16.13.0 o superior. Se instalará sobre un mismo contenedor de docker junto con npm. Se instalará sobre un mismo contenedor de docker junto con NodeJS
   iii. Npm 8.19.1 o superior.

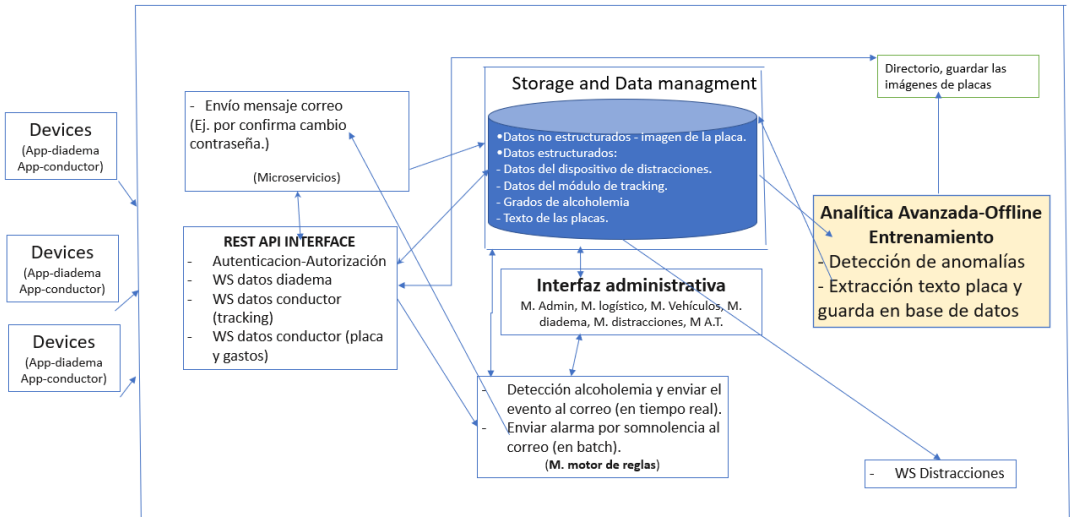
iv. MongoDB 6.0.
v. PHP 7.3 o superior.
vi. Python. Se instalará en el mismo ambiente junto con OpenCV y Tensorflow.
vii. OpenCV
viii. Tensorflow.
ix. Apache web server con SSL. Se conectará con el dominio protégeme.com.co.

La Instancia (VM) será instalada y configurada en Amazon AWS:

- Inicializar nuestra instancia en la nube
- Instalar pilas de software básico y avanzado, según sea necesario
- Agregar perfiles de seguridad a la pila fundamental
- Crear y configurar una base de datos de series temporales
- Dar a nuestra plataforma un nombre de dominio

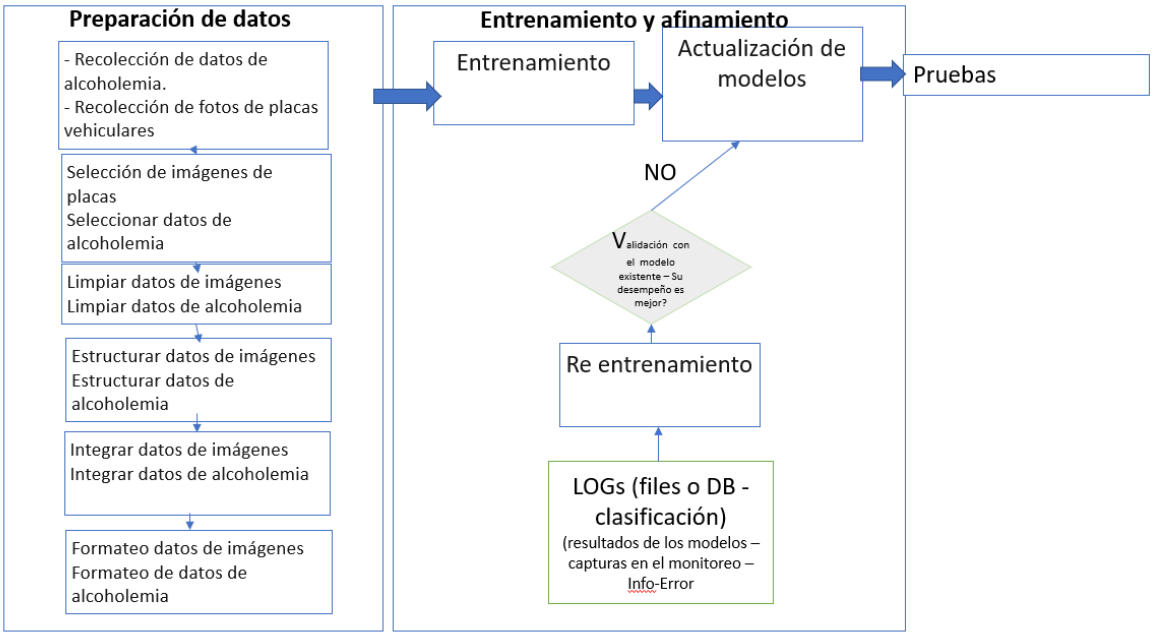
4.2 Despliegue de componentes

En la siguiente figura se muestran los componentes que harán parte de la solución:



Arquitectura funcional de la solución

En este caso, aplica el componente de Analítica Avanzada (offline) Entrenamiento, donde se estarían probando los modelos que serán usados en el ambiente productivo.



Ciclo de vida ML: Alto nivel

El anterior gráfico agrupa el proceso de construcción de los módulos de analítica avanzada en 2 fases:

1. **Preparación de datos:** Primero se recolectan los datos de estado de alcoholemia de la “diadema”, o de las fotos de las placas de los vehículos. Luego de esto se preparan los datos o imágenes para que puedan ser integrados con los modelos de Machine Learning o Deep Learning requeridos para poder realizar las siguientes dos funciones: detección de anomalías o extracción del texto de la placa desde una foto.

2. **Entrenamiento y afinamiento:** Luego de la fase de preparación de datos, un especialista en ciencia de datos con toda la información recolectada selecciona los modelos que puedan lograr los objetivos de detección de anomalías o extracción de texto, luego entrena y afina estos modelos de tal manera que puedan ser aplicados en producción de manera eficiente, es decir pasar a la fase de Despliegue y monitoreo.