Documentación sobre el proyecto “Sistema de Gestión de Flotas Inteligente”

Estudiante:

Celene Parra Vega

Institución Educativa Asamblea Departamental

Problema Complejo de Arquitectura

Medellín – Colombia

Introducción

Se le dará solución de acuerdo a los requerimientos solicitados y lo aprendido en clase.

Desarrollo

En el siguiente trabajo se especificará como se planteó la posible solución.

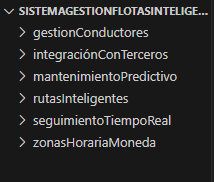
**Figura 1.**

*Funcionalidad Principal:*

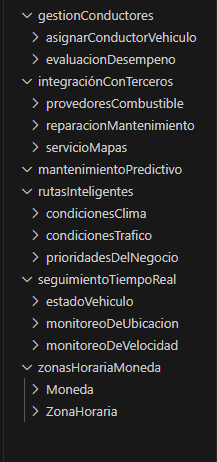
* *Seguimiento en tiempo real: Los administradores deben poder monitorear la ubicación, velocidad y estado de cada vehículo.*
* *Rutas inteligentes: Optimizar las rutas en función de condiciones de tráfico, clima y prioridades del negocio.*
* *Mantenimiento predictivo: Generar alertas para el mantenimiento preventivo basado en sensores de los vehículos.*
* *Gestión de conductores: Asignar conductores a vehículos y evaluar su desempeño (p. ej., consumo de combustible, seguridad).*
* *Integración con terceros: Permitir la conexión con proveedores externos, como servicios de mapas, combustible y reparaciones.*

*.*

Como el proyecto se realizará bajo una arquitectura de ´Monolito granular´ los principales servicios serán cada funcionalidad principal, como se muestra en la siguiente imagen:



Cada monolito tiene unos submódulos con los aspectos más específicos de cada funcionalidad, como se muestra a continuación:

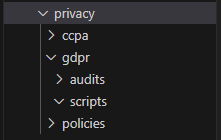


**Figura 2.**

*Requisitos Técnicos:*

* *Escalabilidad: El sistema debe soportar miles de vehículos activos y consultas simultáneas.*
* *Disponibilidad: Alta disponibilidad, con un SLA del 99.9%.*
* *Modularidad: Los módulos del sistema deben ser independientes y permitir fácil mantenimiento.*
* *Almacenamiento eficiente: Manejar grandes volúmenes de datos históricos (rutas, sensores, eventos) sin pérdida de rendimiento.*
* *Seguridad:*
* *Protección de datos personales (conductores, ubicaciones).*
* Cumplir con normativas de privacidad (GDPR y similares).

El proyecto de escogió con una arquitectura granular para que a futuro tenga una estabilidad más fácil y cómoda, mientras tanto para que el sistema soporte se podría tener una base de datos de pantalla o un balanceador de cargas entre los datos esta solución sería mientras el presupuesto sea el adecuado para cambien a una arquitectura de microservicios completa no solo de lógica. Si se cumple con el requisito de SLA y molaridad. En el caso de la modalidad se manifiesta claramente en los microservicios hechos lógicamente en la carpetacion, anuqué los ideal seria los microservicios en totalidad no solo lógicamente de acuerdo al presupuesto es la mejor opción, y en cuanto en el almacenamiento para se usara una DB de MongoDB la cual es de bajo costo y es adecuada para los requerimientos actuales a esto del añade el hecho que para cumplir las normativas de privacidad (GDPR y similares) se creo una carpeta en cada submódulo llamada “privacy” en la cual entre sus carpetas habrá Scripts para anonimización y eliminación de datos innecesarios o de poca redundancia para la empresa, en esta misma carpeta se tendrá el objetivo de la seguridad y protección de datos, la carpeta es la siguiente:



**Figura 3.**

*Restricciones:*

*• El sistema debe permitir la implementación gradual (MVP para logística y luego expansión a transporte público y servicios de entrega).*

*• Múltiples zonas horarias y monedas (tarifas de transporte o costos de mantenimiento).*

*• Presupuesto limitado en la fase inicial.*

*• El cliente quiere poder personalizar partes del sistema según la industria.*

El gráfico proyecto al ser un monolito granular se tiene será mas flexible al momento de quitar o agregar nuevas funcionalidades teniendo en cuenta el presupuesto, para “*Múltiples zonas horarias y monedas*” se creo un microservicio enfocado en esa restricción y al tener una buena organización el proyecto será legible y sostenible para un futuro.

**Figura 4.**

*Los doce factores.*

I. Código base (Codebase)

Un código base sobre el que hacer el control de versiones y multiples despliegues

Se usará el estilo arquitectónico monolito granular para mayor facilidad en la escalabilidad del proyecto, a esto se le usará el código fuente de tipo polirepo en GitLab en el cual habrá un repo principal de todo el proyecto y cada módulo tendrá su propio repo

II. Dependencias

Declarar y aislar explícitamente las dependencias

Para la dependencia cada módulo tiene el archivo “package.json”.

III. Configuraciones

Guardar la configuración en el entorno

Para este punto cada modulo tiene una carpeta llamada “/confit” y dentro de estas en la carpeta “/settings” se tendrán todas las variables de entorno que se necesiten.

IV. Backing services

Tratar a los “backing services” como recursos conectables

Estos se verán en la carpeta “/confit” y/o “/scripts” De acuerdo con la necesidad y se podrá usar para la configuración y conexión de Apis o para las BD.

V. Construir, desplegar, ejecutar

Separar completamente la etapa de construcción de la etapa de ejecución.

Para este punto están las carpetas “/ci-cd” Para las conexiones del repo y “/docker”.

VI. Procesos

Ejecutar la aplicación como uno o más procesos sin estado.

La lógica está en la carpeta “/services”

VII. Asignación de puertos

Publicar servicios mediante asignación de puertos

Esto se podría configurara en la carpeta “/confit” en conjunto con la carpeta “/docker”.

VIII. Concurrencia

Escalar mediante el modelo de procesos.

Si la cumple ya que al ser una arquitectura de monolito granular cada microservicio (módulos) funciona de manera independiente, siendo mas escalable y con su propia carga de trabajo

IX. Desechabilidad

Hacer el sistema más robusto intentando conseguir inicios rápidos y finalizaciones seguras

Si ya que al ser independiente no altera en gran medida a otros.

X. Paridad en desarrollo y producción

Mantener desarrollo, preproducción y producción tan parecidos como sea posible

Si ya que usar carpetas como “/conf”, “/settings” y “/docker” ayuda a mantener consistencia entre entornos y en cada etapa del proyecto.

XI. Historiales

Tratar los historiales como una transmisión de eventos

Si ya que hay carpetas que van dirigidas a esto como la de “/logs”

XII. Administración de procesos

Ejecutar las tareas de gestión/administración como procesos que solo se ejecutan una vez.

Si se cumple ya que estas tareas son independientes.

Nota: La implementación del SOLID esta solo en la carpeta asignarConductorVehiculo.

• ¿Qué es un contenedor en micro servicios?

Un contenedor en microservicios es una unidad que agrupa una aplicación, su configuración y dependencias, y puede implementarse de forma independiente. Los contenedores son una forma de virtualizar sistemas operativos y son ideales para agrupar e implementar microservicios independientes.

* ¿Qué es pipeline?

Se trata de un conjunto de herramientas y procesos que recogen datos de distintas fuentes, los analizan y presentan los resultados de forma comprensible. Los pipelines de datos pueden ayudar a las empresas a tomar decisiones estratégicas basadas en datos reales.