**ARQUITECTURA DE SOFTWARE**

1. **INTRODUCCIÓN**

En este documento se describe la arquitectura de software propuesta para el backend de un sistema de gestión de servicios de transporte. Se utilizará Laravel como framework principal para el desarrollo del backend, junto con Laravel WebSockets para la comunicación en tiempo real entre el servidor y los clientes.

1. **JUSTIFICACIÓN DE LAS HERRAMIENTAS TECNOLÓGICAS**

**Laravel:** Se elige Laravel como el framework principal debido a su elegancia, facilidad de uso y robustez. Laravel proporciona una arquitectura MVC bien definida que permite una estructura clara y organizada del código, lo que facilita el mantenimiento y la escalabilidad del proyecto. Además, Laravel viene con una amplia gama de características integradas, como el sistema de enrutamiento, el ORM Eloquent, las migraciones de base de datos y la autenticación, que aceleran el desarrollo de la aplicación.

**Laravel WebSockets**: Se utiliza Laravel WebSockets para la comunicación en tiempo real entre el servidor y los clientes. Esta herramienta proporciona una solución eficiente y escalable para implementar WebSockets en aplicaciones Laravel. Permite la creación de servidores WebSocket con facilidad y proporciona una API sencilla para enviar y recibir mensajes en tiempo real. Esto es fundamental para la funcionalidad de seguimiento en tiempo real del conductor y la notificación instantánea de eventos importantes en la aplicación.

**MySQL:** Se utiliza MySQL como sistema de gestión de base de datos debido a su amplia adopción, estabilidad y rendimiento. MySQL es compatible con Laravel y ofrece una amplia gama de características que facilitan el almacenamiento y la manipulación de datos. Además, MySQL es una opción económica y de código abierto, lo que lo hace ideal para proyectos de cualquier tamaño.

1. **ESTRUCTURA DE LA APLICACIÓN**

La aplicación seguirá una arquitectura MVC (Modelo-Vista-Controlador) para mantener una separación clara de las preocupaciones y facilitar la escalabilidad y el mantenimiento del código.

* 1. Modelo (Model)

Los modelos en Laravel representarán los recursos de la API, como conductores, usuarios y servicios. Estos modelos definirán la estructura de los datos y las relaciones entre los recursos.

Puedes usar las migraciones de Laravel para definir la estructura de la base de datos y el ORM Eloquent para interactuar con los modelos y realizar operaciones CRUD en la base de datos.

* 1. Controlador (Controller)

Los controladores en Laravel serán responsables de manejar las solicitudes HTTP entrantes y orquestar las acciones necesarias en la aplicación.

Los métodos de los controladores responderán a las diferentes rutas de la API y realizarán operaciones como crear, leer, actualizar y eliminar recursos.

* 1. Rutas (Routes)

Las rutas en Laravel definirán las URL y los métodos HTTP que se utilizarán para acceder a los diferentes recursos de la API.

Se Puede usar las rutas de Laravel para vincular las solicitudes HTTP entrantes a los métodos correspondientes en los controladores.

* 1. Middleware

Los middleware en Laravel se pueden utilizar para aplicar lógica de procesamiento adicional a las solicitudes HTTP entrantes antes de que lleguen a los controladores.

Puedes usar middleware para realizar la autenticación, la autorización, la validación de datos y otras tareas de procesamiento antes de que se procesen las solicitudes.

1. **FLUJO DE DATOS Y COMUNICACIÓN EN TIEMPO REAL**

**Solicitudes HTTP:** Las solicitudes HTTP se utilizarán para operaciones CRUD estándar, como crear un nuevo servicio o actualizar la ubicación de un conductor. Estas solicitudes se manejarán mediante rutas definidas en el archivo de rutas de Laravel y serán dirigidas a los controladores correspondientes para su procesamiento.

**WebSockets:** Laravel WebSockets se utilizará para la comunicación en tiempo real entre el servidor y los clientes. Esto permitirá la actualización en tiempo real de la ubicación del conductor y la notificación instantánea de eventos importantes, como la cancelación de un servicio. Los eventos emitidos por el servidor se capturarán en el cliente mediante la implementación de suscriptores de WebSocket.

1. **CONCLUSIÓN**

La arquitectura propuesta aprovecha las capacidades de Laravel y Laravel WebSockets para proporcionar una solución robusta y escalable para el backend del sistema de gestión de servicios de transporte. El uso de Laravel permite un desarrollo rápido y eficiente, mientras que Laravel WebSockets facilita la implementación de comunicación en tiempo real. MySQL proporciona una base de datos confiable y de alto rendimiento para almacenar y manipular datos. En conjunto, estas herramientas tecnológicas forman una arquitectura sólida que cumple con los requisitos del proyecto y sienta las bases para una aplicación exitosa y fácil de mantener.