El proyecto debido a su utilidad presentará una alta concurrencia de peticiones, se puede presentar el caso en el que un usuario solicite un servicio y dos conductores acepten el servicio al mismo tiempo, lo cual desemboca en una doble asignación hacia un solo servicio, ¿Como se puede evitar este tipo de problemas, para garantizar que un servicio sólo podrá ser atendido por un solo conductor? (Ayuda: revisar el tema de condición de carrera)

R:: Para evitar el problema de la condición de carrera y garantizar que un servicio solo pueda ser atendido por un solo conductor, se pueden implementar varias estrategias como lo son:

* Transacciones en la Base de Datos (implementada en el código)
* Bloqueo de Recursos
* Atomicidad en las Operaciones
* Verificación de Estado (implementada en el código)
* Uso de Colas de Trabajo

Suponga que se requiere crear un tercer módulo web donde solo

se quiere conocer las ubicaciones en tiempo real de los usuarios

conductores sobre un mapa; en este mapa se desea manejar dos

iconos con dos estados para los usuarios conductores:

**Activo:** Donde el conductor está emitiendo coordenadas constantemente

**Inactivo:** Donde el conductor ha dejado de emitir coordenadas después de 1 minuto.

¿Cómo puede usted plantear una solución para poder mostrar el estado inactivo en este módulo?

R::

**Implementación de Estado Inactivo para Conductores en Tiempo Real**

Para abordar la necesidad de mostrar el estado inactivo de los conductores en tiempo real en un módulo web, se puede diseñar una solución que aprovecha las capacidades de Laravel y MySQL.

Paso 1: Registro de Última Actualización

En la tabla de drivers de nuestra base de datos MySQL, hemos agregado un campo llamado last\_location para almacenar la fecha y hora de la última actualización de ubicación del conductor.

Paso 2: Proceso de Actualización Periódica

Se puede crear un comando personalizado de Artisan en Laravel, llamado UpdateDriverStatus, programado para ejecutarse cada minuto utilizando las herramientas de programación de tareas de Laravel.

Paso 3: Determinación del Estado del Conductor

Dentro del comando UpdateDriverStatus, se compara la fecha y hora de la última actualización de ubicación de cada conductor con el tiempo actual. Si la diferencia es mayor que 1 minuto, el conductor se marca como inactivo.

Paso 4: Actualización del Estado en la Base de Datos

El comando UpdateDriverStatus actualizaría el estado del conductor en la base de datos según los resultados de la verificación. Si la diferencia es menor o igual a 1 minuto, el conductor se marca como activo; de lo contrario, se marca como inactivo.

Paso 5: Visualización del Estado en el Módulo Web

En el módulo web que muestra las ubicaciones en tiempo real de los conductores, consulto el estado de cada conductor desde la base de datos y muestro el icono correspondiente en el mapa. Se utilizaría diferentes iconos o colores para representar los estados activo e inactivo.

Paso 6: Actualización en Tiempo Real (Opcional)

Para proporcionar una experiencia más dinámica, se podría integrar el uso de WebSockets en Laravel. Esto nos permite recibir actualizaciones en tiempo real del estado de los conductores desde el servidor y actualizar dinámicamente los iconos en el mapa sin necesidad de recargar la página.