2. CODE REFACTORING

Malas Prácticas	Correcciones
Nombre de la función "**post_confirm**" se hace con notación de underscore no muy practicado por la comunidad javascript	Se utiliza la notación lowerCamelCase para los nombres variables y funciones y CAPITAL_UNDERSCORE para las constantes.
La variable "**servicio**" es escrita en español	Se cambia el nombre de la variable "servicio" por "**service**" en ingles
La variable "**service**" no existe en la función	
Condiciones "**servicio != NULL**", "**servicio.driver_id == NULL**" y "**servicio.user.uuid == ''**" como forma para verificar si una variable es NULL o un **string** vacío son inadecuadas en javascript	Se definen las condiciones de "**service**", "**service.driver_id**" y "**service.user.uuid**" con solo la variable para identificar si es NULL o string vacio.
Dejar código comentado en el script	Se eliminan los comentarios los cuales son innecesarios
Comentario "Notificar a usuario!!" está en español y no es adecuado	
No se define en una variable constante el valor para "**params.driver_id**"	Se define un valor constante "**driverId**" para el valor "**params.driver_id**
Se guarda en la variable "servicio" el método "**Service.update**" sin ningún objetivo claro	Se elimina la variable servicio donde se asignaba al método "**Service.update**"
Se sobrescribe la variable servicio nuevamente con el mismo dato "servicio = **Service.find(id)**" dentro del bloque if	Se elimina el código que sobrescribe con el mismo valor "**Service.find(serviceId)**" a la variable "**service**"
La condición **(servicio.user.uuid == '')** está mal posicionada ya que su objetivo es condicionar la notificación push	Se define la condición "**if (service.user.uuid)**" para ejecutar una notificación push y el retorno del error cero se ejecutaría igualmente

No se define el alcance (let, var) de las variables "**driverTmp**" y "**push**"	Se define la variable "**driverTmp**" como "let" ya que tiene alcance solo en su bloque y también la variable **pushMessage**
El alcance de "**pushMessage**" no es necesario que sea del tipo "var"	
Se ejecuta dos veces la función "**Service.update**" innecesariamente para incluir otro dato llamado "**car_id**"	Se ejecuta una sola vez la función "**Service.update**" incluyendo el valor para "**car_id**"
Los tipos de error y los valores con que se comparan "**servicio.status_id**" y "**servicio.user.type**" no están definidos por lo cual hace difícil la lectura de estos valores	Se definen en constantes los tipos de error, los id de status de servicio y los tipos de usuario de servicio.
Gran número de líneas de condigo para la función	Se crea la función "**notifyToUser**" para ejecutar la notificación push

3 PREGUNTAS

1. ¿En qué consiste el principio de responsabilidad única? ¿Cuál es su propósito?

Consiste en que cada módulo de la aplicación debe tener una responsabilidad especifica que no es repetida en otro modulo del software que se desarrolla. Tiene el propósito de lograr un nivel de cohesión elevado entre los módulos del software para que se puedan obtener muchos beneficios como escalabilidad, mantenimiento, legibilidad, entre otros.

2. ¿Qué características tiene según su opinión “buen” código o código limpio?

1. Las variables y funciones deben tener nombres que faciliten entender el objetivo por el cual fueron creadas.

2. Debe estar correctamente indentado por cada bloque que se necesite en el código
3. Reutilización de bloques o segmentos de código si son necesarios en varios lugares.
4. Documentación de código a través de comentarios si es necesario para explicar lógica o funcionalidad para la revisión de un programador