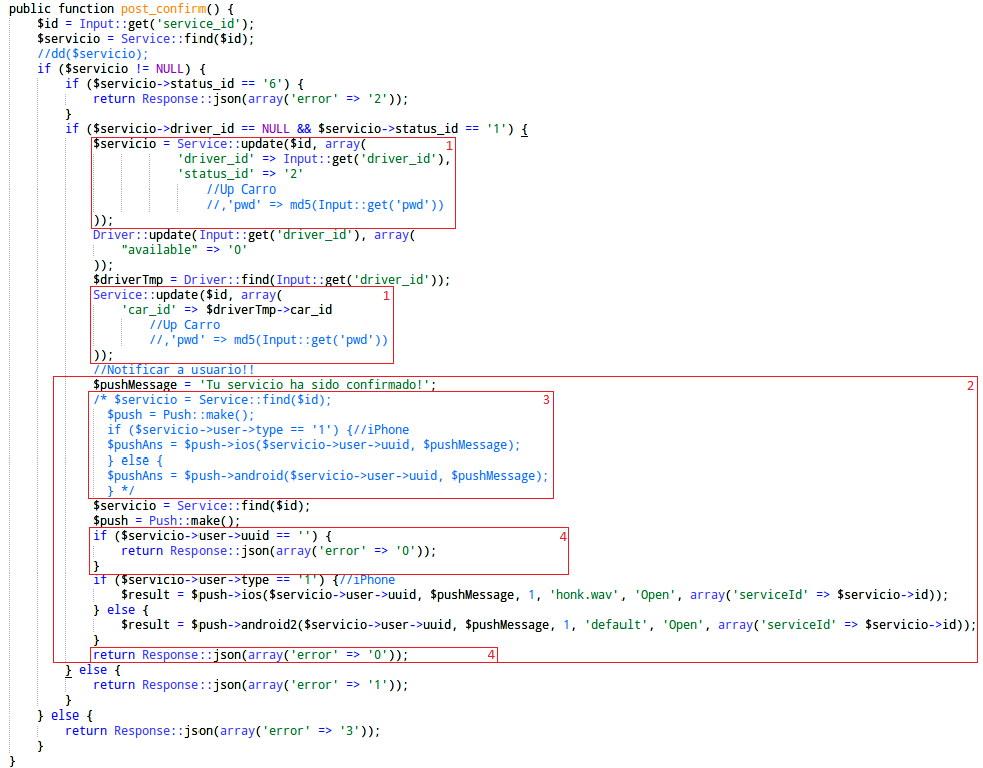
**Code Refactoring**

1. Código anterior



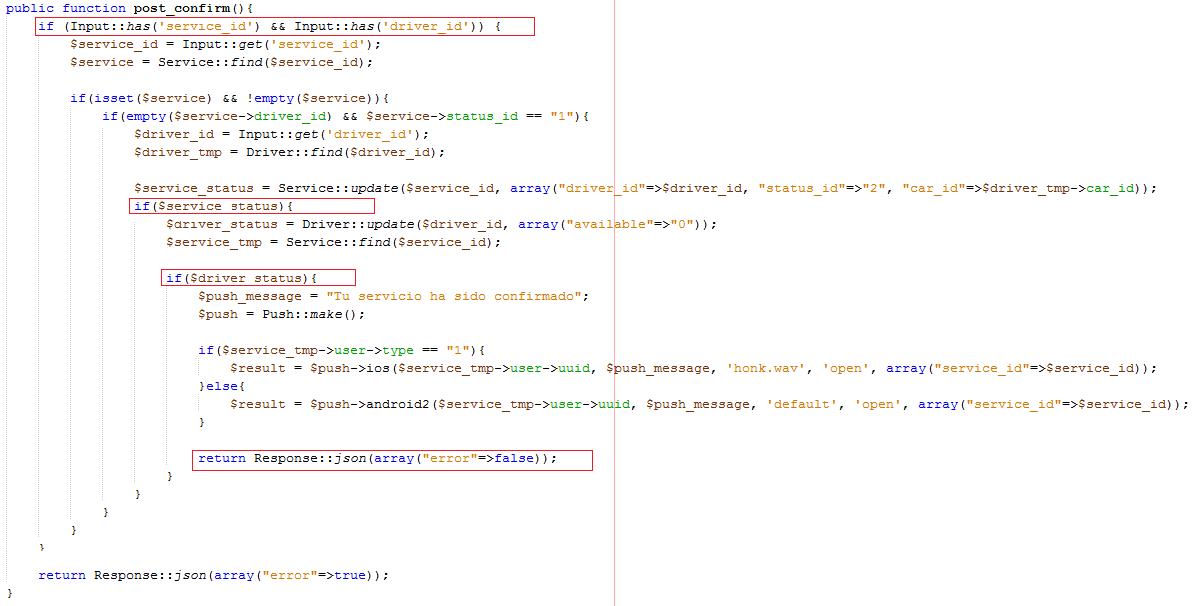
Como se muestra en la imagen anterior, el código presenta diferentes implementaciones inadecuadas, las cuales son:

* Código repetido: se puede evidenciar que inicialmente se realiza una actualización del servicio y posterior a la actualización del conductor se vuelve a actualizar el servicio como confirmado, lo ideal sería que primero se confirme el conducto y posteriormente se actualice el servicio para realizar una sola sentencia SQL.
* Propenso a Bugs: no se está validando si se realizaron de manera satisfactoria la actualización de “Driver” y “Service” por lo que sin importar que esto suceda, siempre se returna una notificación y un estado “error = 0”

De igual manera tampoco se está validando si los parámetros “service\_id” y “driver\_id” existen y no están vacíos.

* Código sin implementar: existen funciones desarrolladas que no se usan y por ende están comentadas
* Validaciones innecesarias: se tiene un condicional que verifica “$service->user->uuid” para devolver un valor que sin importar que el condicional se cumpla de igual manera se efectúa.

1. Código Optimizado



Como lo dice la descripción del código, el objetivo de este es cambiar el estado del conductor y servicio al igual que enviar una notificación y una respuesta, generalice el código de tal manera que no se envíe un código de error en cada paso, pues lo importante es resaltar si el proceso se realizó o no satisfactoriamente.

Como se muestra en la imagen anterior las malas prácticas de programación quedan resueltas, debido a que inicialmente se verifica que los parámetros que se utilizan en las actualizaciones de “Driver” y “Service” existan, posteriormente se verifica que las actualizaciones de “Driver” y “Service” se hayan efectuado, al igual de que la actualización de “Service” se realiza una vez actualizado “Driver”.