PRUEBA TÉCNICA PARA EL BACKEND DE RAPPI

Lee la prueba y teniendo en cuenta tu disponibilidad de tiempo, infórmanos el día y hora en la que máximo enviarás las respuestas. Nota que en el transcurso del tiempo podrás enviar preguntas que serán respondidas con la mayor prontitud. Si por cualquier motivo no puedes terminar toda la prueba en el tiempo que estipulado, envía las partes que tengas y el motivo por el cual no pudiste completarla. Una adición al tiempo de entrega podría ser otorgada.

CODING CHALLENGE (70 puntos)

La primera parte de la prueba consiste en un Coding Challenge tomado de Hackerrank. Lo puedes encontrar en el siguiente Link. Puedes escribir el programa en el lenguaje de programación en el que te sientas más cómodo, aunque se prefiere el uso de javascript o PHP (En caso de ser PHP también es preferible el uso de Laravel. Puedes usar cualquier mecanismo para la entrada y salida de datos del programa. Una vez hayas terminado el código describa en un documento brevemente:

1. Las capas de la aplicación (por ejemplo capa de persistencia, vista, de aplicación, etc) y qué clases pertenecen a cual
2. La responsabilidad de cada clase creada

Recibirás puntos extra si:

1. El mecanismo usado para la entrada y salida de datos es Web (5 puntos)
2. Usas Git o SVN y la historia del programa muestra su progreso en el desarrollo, usando commits con unidades de funcionalidad (5 puntos)
3. Usas pruebas unitarias y/o de validación (acceptance) que demuestren el buen funcionamiento del programa (10 puntos)

CODE REFACTORING (20 puntos)

El siguiente código muestra el método de un controlador que:

1. Recibe dos parámetros por POST: El id de un servicio, el id de un conductor
2. Cambia el estado de varias entidades en la base de datos basado en la lógica del negocio.
3. Envía notificaciones y retorna una respuesta.

Refactorice y envíe el código y en un documento explique:

1. Las malas prácticas de programación que en su criterio son evidenciadas en el código
2. Cómo su refactorización supera las malas prácticas de programación

```
public function post_confirm() {
    $id = Input::get('service_id');
    $servicio = Service::find($id);
    //dd($servicio);
    if ($servicio != NULL) {
        if ($servicio->status_id == '6') {
            return Response::json(array('error' => '2'));
        }
        if ($servicio->driver_id == NULL && $servicio->status_id == '1') {
            $servicio = Service::update($id, array(
                        'driver_id' => Input::get('driver_id'),
                        'status_id' => '2'
                            //Up Carro
                            //,'pwd' => md5(Input::get('pwd'))
            ));
            Driver::update(Input::get('driver_id'), array(
                "available" => '0'
            ));
            $driverTmp = Driver::find(Input::get('driver_id'));
            Service::update($id, array(
                'car_id' => $driverTmp->car_id
                    //Up Carro
                    //,'pwd' => md5(Input::get('pwd'))
            ));
            //Notificar a usuario!!
            $pushMessage = 'Tu servicio ha sido confirmado!';
            /* $servicio = Service::find($id);
              $push = Push::make();
              if ($servicio->user->type == '1') {//iPhone
              $pushAns = $push->ios($servicio->user->uuid, $pushMessage);
              } else {
              $pushAns = $push->android($servicio->user->uuid, $pushMessage);
              } */
            $servicio = Service::find($id);
            $push = Push::make();
            if ($servicio->user->uuid == '') {
                return Response::json(array('error' => '0'));
            }
            if ($servicio->user->type == '1') {//iPhone
                $result = $push->ios($servicio->user->uuid, $pushMessage, 1, 'honk.wav', 'Open', array('serviceId' => $servicio->id));
            } else {
                $result = $push->android2($servicio->user->uuid, $pushMessage, 1, 'default', 'Open', array('serviceId' => $servicio->id));
            }
            return Response::json(array('error' => '0'));
        } else {
            return Response::json(array('error' => '1'));
        }
    } else {
        return Response::json(array('error' => '3'));
    }
}
```

PREGUNTAS (10 puntos)

Responde y envía en un documento las siguientes preguntas:

1. ¿En qué consiste el principio de responsabilidad única? ¿Cuál es su propósito?
2. ¿Qué características tiene según tu opinión "buen" código o código limpio?