Coding challenge

1. Es una aplicación que no requiere mayor almacenamiento ni persistencia, por lo que los datos (el cubo) se manejaron en memoria. Tenemos dos capas, la de presentación y la de lógica. En la capa de presentación tenemos el js coms encargado de recibir los datos del usuario por la interfaz, interpretarlos y hacerle las peticiones a la clase de lógica que en este caso solo es el js cube.
2. Coms tiene la responsabilidad de la comunicación entre la interfaz y la lógica. Recoge los datos, los procesa, realiza peticiones a la lógicay despliega respuesta al usuario otra vez. Cube representa la lógica, manejando el cubo. Lo crea y ofrece las funciones para manipularlo.

Code refactoring

Malas prácticas: La utilización de valores específicos sobre el código. Los datos relevantes de negocio se deberían manejar en un archivo de configuración aparte con llaves y valores. Cosas como if(status == 4) no dejan que el código sea legible y además, ante un posible cambio del valor de las variables, lo vuelve un problema para mantener el código.

El manejo de código comentado. Si bien en el momento de realizar el código vale la pena mantener código comentado frente a un determinado cambio, al realizar un comit esto se debería borrar debido a que genera basura visual y dificulta el entendmiento. Si se necesita regresar, se utiliza el sistema de versiones.

Tener tantos retornos puede llegar a resultar peligroso. No se maneja con certeza el fin de las funciones y lo hace dificil de depurar.

En ocasiones atajos para hacer el código corto lo hace menos legible y es preferible gastar más líneas de código pero que se entienda. Cosas como hacer un push sobre un arreglo haciendo llamado automático de un método.

Preguntas

Responsabilidad única: En otros lados llamado técnica del maestro. Consiste en desglosar los problemas hasta poderlo convertir en problemas muy pequeños que puedan dividirse por responsabilidad. Con esto se pueden crear clases con responsabilidades pequeñas y concretas. La unión de todas las clases y funciones resultan en la resolución del problema completo. Como una operación matemática complicada. Mejor se divide en pequeña operaciones que en conjunto dan el resultado completo.

Código limpio: Documentado, sin redundancia de variables, sin zonas muertas (combinaciones lógicas que dejan código inaccesible i.e. if(false) …). Nombramiento seguido de estándares y con nombre diciente (de métodos y funciones). Indentado para reconocer las secciones del código. Evitar espacios innecesarios pero dejarlos suficientes para dividir el código. Uso de constantes para valores del negocio.