**PRUEBA BACKEND RAPPI:**

**CODING CHALLENGE**

1. Capas:

* **Aplicación**, las siguientes clases:
  + **matrix3DController:**

recibe peticiones fuera del sistema de aplicación y devuelve los objetos del modelo que se requiera (Patron Controller, GRASP).

* + - getSetTestCase: Obtiene un objeto setTestCase.
    - getOutputFormat: Obtiene un objeto outputFormat
  + **setTestCase**
    - Representa un conjunto de casos de prueba, es un objeto del modelo real.
    - Su responsabilidad es procesar todo ese conjunto de pruebas.
  + **testCase**
    - Representa una prueba, es también traída del modelo real.
    - Su responsabilidad es procesar una sola prueba
  + **operation:**
    - Representa una operación de actualización o suma, tiene dos subclases a su vez que son:

updateOperation y queryOperation, mediante polimorfismo, se le da una nueva escritura o implementación a la responsabilidad de procesar una operación.

* **Persistencia,** las siguientes clases:
  + **inputFormat**

Se encarga de convertir una entrada de archivo o texto a una representación de instrucciones; hay una subclase para cada implementación (inputFile, inputString).

* + **linesInput**

Se encarga de construir un objeto de instrucciones, donde está toda la información que necesitará la capa de aplicación para procesar.

* **Vista,** las siguientes clases:
  + **outputFormat**

Se encarga de reunir las instrucciones de suma y mostrar los resultados

**CODE REFACTORING**

**\*** PHP

**PREGUNTAS**

1. El principio de responsabilidad única, viene de los patrones GRASP, y se refiere hay que preguntarse si una responsabilidad de esa clase es verdaderamente únicamente de ella o, si debe recibir colaboración de otras clases para su implementación.
2. Seguir los patrones GRASP, proteger el código para prever variaciones, que haya una cantidad moderada de acoplamiento entre clases, uso de interfaces para la comunicación con otros sistemas.