Entrega 4 Parte II

Justificación funcionalidad de API Rest

Con respecto a la API de Vinculación decidimos utilizar SpringBoot para levantar la api. SpringBoot tiene la particularidad de tener una clase RestController en la cual cada método va a entender un verbo HTTP. En nuestro caso, se utiliza POST para hacer uso de la funcionalidad de nuestra api que es “EjecutarVinculacion”. La clase RestController que manejamos es: VinculacionController.

Todos los datos que van desde Gesoc a la API son convertidos a tipos de datos Gson mediante unos métodos definidos que tiene la clase GsonConverter y la API también tiene una clase Gson la cual se encarga de transformar estos datos Gson en las clases específicas que representan cada uno. Esto conlleva a que la API entienda las clases OperaciónDeIngreso y OperaciónDeEgreso.

OperacionDeEgreso tiene los atributos de las anteriores entregas y además se le agregan la referencia a una operacionDeIngreso mediante su ID y un booleano para saber si ya tiene una vinculación asociada o no. De la mano con esto, tiene un método nuevo: asociarOperacionDeIngreso el cual se encarga de cargar este id y actualizar este valor booleano.

OperaciónDeIngreso tiene los atributos de las anteriores entregas y además una lista de OperacionDeEgreso, un monto sin vincular (que se actualiza su valor a medida que se vinculan egresos) y dos fechas: Una máxima y una mínima las cuales son las fechas límite para que un egreso pueda ser asociado a este ingreso. Sirven para indicar el periodo de asociación aceptable que tiene un ingreso con un egreso. De la mano con esto, tiene un método nuevo: asociarEgreso el cual se encarga de agregar a la lista de egresos la operación que se asocia y actualiza el montoSinVincular.

Para poder asociar dos operaciones se deben cumplir ciertas características. Para resolver esto planteamos una clase Validacion la cual tiene un método puedenVincularse(egreso, ingreso) y te devuelve un booleano dependiendo de si cumplen con los requisitos o no.

Hay un caso particular en el CriterioFecha en el cual se agrega una condición más a las condiciones para poder validar una vinculación por lo cual planteamos una clase ValidacionPeriodo la cual hereda de la clase Validación el método de puedenVincularse() y además evalúa otras condiciones especificas de esta clase ValidacionPeriodo para devolver el resultado booleano.

Por otro lado para resolver la problemática de los Criterios de Ejecucion planteamos una interfaz CriterioEjecucion la cual tiene el método ejecutarse(egresos, ingresos) y devuelve una lista de ingresos modificada con formato Gson.

Todos los criterios implementan esta interfaz además de heredar de una clase VinculacionService la cual tiene cierta funcionalidad compartida entre los diferentes criterios. Sobre todo la parte de asociar egresos con ingresos lo cual lo hacen todos los criterios de una u otra manera. Esta clase VinculacionService hereda de la clase Validacion ya que hace uso del método puedenVincularse para realizar las asociaciones.

Por el lado de Gesoc; en la clase ServicioVinculacionEgresosIngresos llamamos mediante retrofit a la API para que nos haga la funcionalidad que nosotros queremos mediante el método ejecutarVinculación(ingresos, egresos, criterios).

Este método hace uso de la interfaz VinculacionService la cual tiene un método POST que hace un Call ejecutarCriterio(egresos, ingresos, criterios) con los datos en formato Gson para que la API pueda trabajar con ellos.