

GeSoc

Decisiones de diseño

**Entrega 1**

[**Introducción**](#_pvrdgv4k41hh) **2**

[**Egresos**](#_ifuqisyoebvi) **3**

[**Medios de pago**](#_xqcfo9l230pf) **5**

[**Documentos comerciales**](#_og8nvse0ju34) **5**

[**Proveedores**](#_qd4x4j7l3rlp) **6**

[**Organizaciones y entidades**](#_hcgvttw47g41) **7**

[**Usuarios**](#_5fhl7zpsij44) **8**

# Introducción

En esta primera entrega de GeSoc, apuntamos a crear todas las entidades que cubran el 100% de los requerimientos propuestos por la cátedra.

En muchos de los requerimientos, encontramos algunos vacíos en cuanto a los comportamientos esperados para cada entidad. Entendemos que con las sucesivas entregas, se irá dotando de comportamiento a cada una de esas entidades mencionadas.

No obstante, preferimos aplicar en cada caso la alternativa más sencilla (KISS) evitando estructuras que requieran un retrabajo mayor en las próximas entregas.

# Egresos

Es una de las abstracciones más importantes del dominio, dado que, para esta primera entrega, concentra la relación entre gran parte de nuestras clases.

Para esta abstracción, pensamos inicialmente en una herencia: una clase Abstracta “Operación”, de la que hereden “Egresos” e “Ingresos” (se menciona la operación de ingresos en el alcance de la aplicación). Sin embargo, y como no pudimos definir qué datos comunes tendrán estas dos subclases, decidimos diseñar una sola abstracción Egresos.

En cuanto a la “construcción” de objetos “Egresos”, incluímos un constructor que obligue a enviar como parámetro los siguientes atributos:

* Fecha
* Proveedor
* Organización
* Medio de pago

Existirán dos métodos adicionales (del tipo Setter) que permitirán:

* Agregar un documento comercial asociado (no obligatorio).
* Agregar ítems detalle al egreso.

Respecto a los ítems del egreso, estos tendrán cada uno:

* El producto, que decidimos implementarlo como un String, dado que no tenemos información alguna de la entidad.
* El precio.

# Medios de pago

Modelamos los Medios de pago, con dos atributos:

* Código
* Tipo de medio de pago

El código contendrá el identificador del medio de pago. El tipo de medio de pago podrá ser alguna de las opciones incluidas en el Enum correspondiente.

# Documentos comerciales

Aún no conocemos el comportamiento que pudiera tener cada uno de los documentos comerciales. Definimos implementarlo con un enum.

# Proveedores

La abstracción contiene el identificador del proveedor (CUIT / CUIL), su denominación (razón social / nombre y apellido) y la dirección.

La dirección tiene su propia clase, que contiene los siguientes atributos:

* Calle
* Numero
* CPA
* DatosAdicionales
* DireccionPostal

# Organizaciones y entidades

Siendo que las entidades pueden ser “intercambiables” a lo largo del sistema, entendemos conveniente crear una Interfaz “Organizacion” que, en principio, entienda el mensaje “cargarEgreso”.

De esta forma, Entidad base, y Entidad jurídica serán clases que cumplen con la interfaz “Organización”.

La organización conocerá sus egresos. Y los egresos tendrán entre sus atributos la organización a la que pertenecen.

Con el fin de validar que una entidad base pueda pertenecer a una, y sólo una entidad jurídica, habrá una doble relación entre estas entidad: la entidad jurídica conocerá a la entidad base asociada, y viceversa.

# Usuarios

Compuesto por usuario, password y un flag que indica si es o no Administrador.

La password recibe diversas validaciones para la generación del usuario:

* No debe encontrarse entre el TOP 10.000 de passwords conocidas.
* Debe tener 8 o más caracteres.
* No debe contener secuencias de caracteres de 3 o más caracteres (‘abc’, ‘123’, ‘abc123’).
* No debe coincidir con el usuario, o alguna palabra del contexto de la aplicación.
* No debe tener el mismo carácter repetido.

Todas estas validaciones se crearon en dos clases que reciben el usuario y la password (atributos de Usuario) y validan que la contraseña sea válida.