




## **Práctica Integradora - Modelado de Datos & Persistencia de Datos**

### **Modelado de Datos**

1. Leer el dominio asignado al equipo.
2. Diseñar y comunicar la solución mediante un modelo de datos del punto anterior para poder persistir en una base de datos relacional. Indicar entidades con sus respectivos campos, tipos de datos, claves primarias, las foráneas, cardinalidad, modalidad y las restricciones según corresponda.
3. Detallar las decisiones más significativas tomadas en el modelo del punto anterior, indicando a su vez si tuvieron casos de desnormalización y por qué.

### **Enunciados**

- Enunciado 1: *Tinderizr* -  Final DDS 20230211.pdf
- Enunciado 2: *Lentti* -  Final DDS 20200222.pdf
- Enunciado 3: *Cooperativa de Internet Rural* -  Final DDS 20220305.pdf

### **Persistencia de Datos (desde el modelo de clases hacia el modelo de datos)**

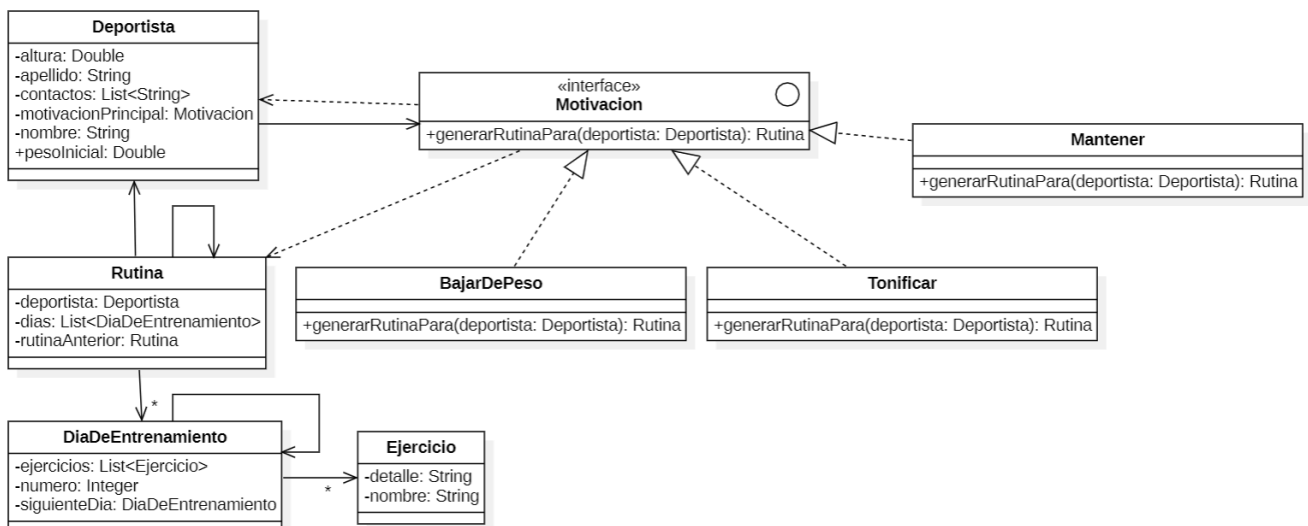
1. Leer el dominio asignado al equipo.
2. Implementar la persistencia del modelo planteado, en Java, con Hibernate como ORM. Clonar el [repositorio base](#) que contiene las clases modeladas.
3. Generar el Modelo de Datos a partir del modelo persistido.
4. Detallar las decisiones más significativas tomadas al realizar los mapeos correspondientes, indicando qué elementos del modelo es necesario persistir y cómo resolvieron los impedance mismatches.

## Enunciados

### Enunciado 1: Generación de Rutinas

Nos han encargado realizar la persistencia de una parte de un Sistema para una cadena de gimnasios. Los deportistas contarán con varios medios de contactos, los cuales serán números de WhatsApp. También escogerán una motivación principal para entrenar, las cuales pueden ser bajar de peso, tonificar el cuerpo o mantener la figura.

De esta motivación dependerá la generación de la rutina que el Sistema generará de forma personalizada para esa persona. La rutina, a su vez, tendrá el detalle por cada día de entrenamiento, cada uno de los cuales contará con los ejercicios a realizar. Es necesario que cada rutina sepa cuál fue su anterior, y que cada día de entrenamiento sepa cuál es el siguiente. Nuestro equipo ya estuvo trabajando y generó el siguiente diagrama de clases que, por cierto, consideramos correcto:



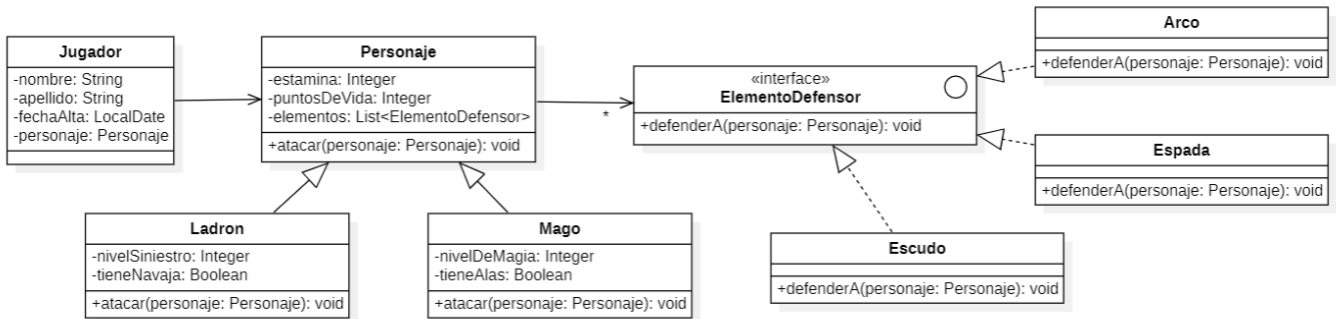
## Enunciado 2: Gloranta

Nos han solicitado el diseño y desarrollo de un juego de roles.

Los distintos jugadores pueden jugar y poseer un solo personaje a la vez.

Los personajes a contemplar en esta primera iteración son los ladrones y los magos. Se sabe que cada uno de estos personajes ataca de una forma diferente a sus oponentes y que poseen diferentes cualidades. Además, los jugadores pueden acumular distintos elementos defensores para usarlos cuando su personaje sea atacado. Cada elemento defensor se comporta de una forma diferente y se sabe que no posee ninguna característica que lo deteriore en el tiempo.

Nuestro ya estuvo trabajando y generó el siguiente diagrama de clases que, por cierto, consideramos correcto:



### Enunciado 3: Módulo de Stock

Nos han solicitado el diseño y desarrollo de un módulo que se encargue de calcular el stock de productos de cualquier local comercial, así como también el cálculo de precios de los mismos.

Se debe tener en cuenta que no solamente existen y se venden productos simples, sino que también existen combos. Un combo está formado por productos que se venden juntos. También pueden existir combos de combos.

Por ejemplo, un local de motos vende motos, cascos, guantes, chalecos, pilotos, entre otros productos; pero también vende algunos combos como por ejemplo guantes + casco + chaleco.

Cada producto tiene un precio en particular. El precio del combo es el resultado de la suma de los productos que contiene.

Además, se debe permitir aplicar descuentos a los distintos productos/combo, los cuales podrían ser acumulables.

Por último, nos han avisado que los productos/combo podrían ser vendidos con distintos packagings, cada uno de los cuales tiene un precio particular que se debería sumar al precio final del producto.

Nuestro equipo ya estuvo trabajando y generó el siguiente diagrama de clases que, por cierto, consideramos correcto:

