# Exercise: Hibernate Code First - Shampoo Company

This document defines the **exercise assignments** for the ["Databases Advanced – Hibernate" course @ Software University.](https://softuni.bg/courses/databases-advanced-hibernate)

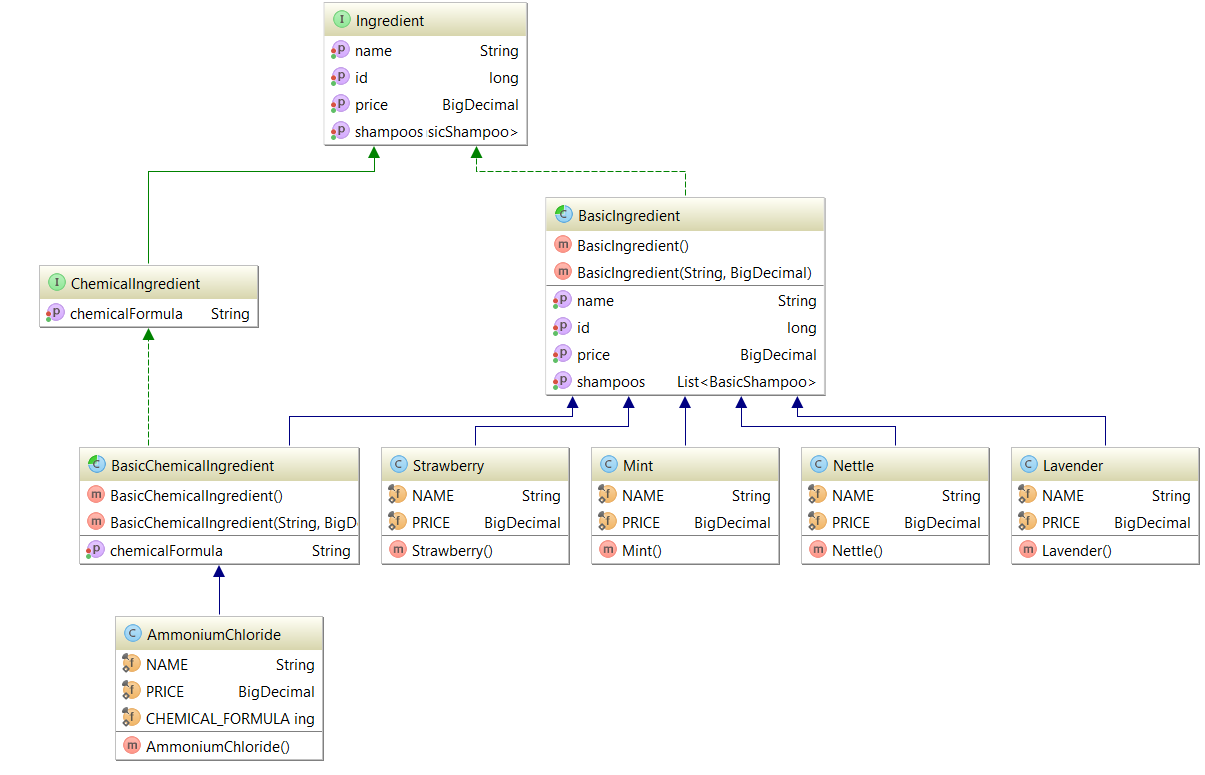
Your task is to create a shampoo company hierarchy. The company produces shampoos, which have **certain set of ingredients**, **label**, **price** and **brand**.

There are several types of shampoos:

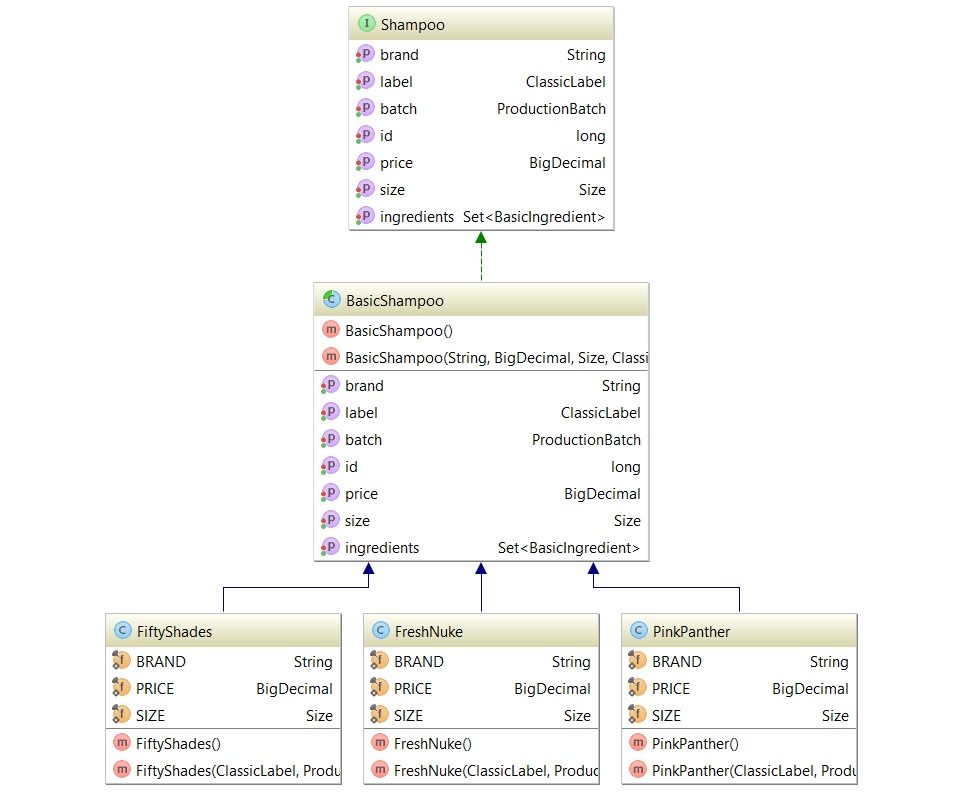
* **Fifty shade**
  + brand: “Fifty Shades”
  + price: 6.69
* **FreshNuke**
  + brand: “Fresh Nuke”
  + price: 9.33
* **Pink Panther**
  + brand: “Pink Panther”
  + Price: 8.50

Each shampoo can be either size BIG, SMALL or MEDIUM.

The ingredient structure should look like this:



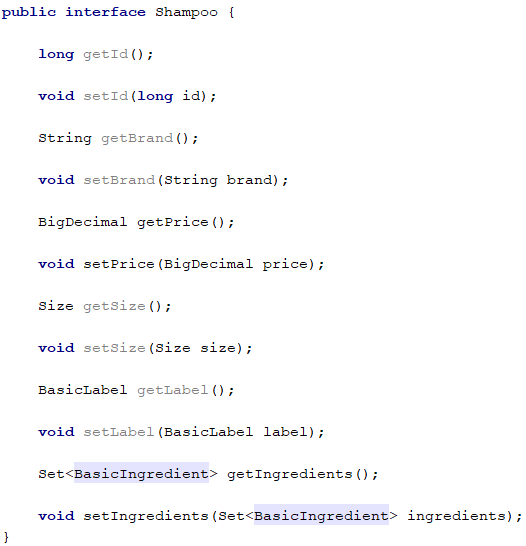
The ingredient structure should look like this:



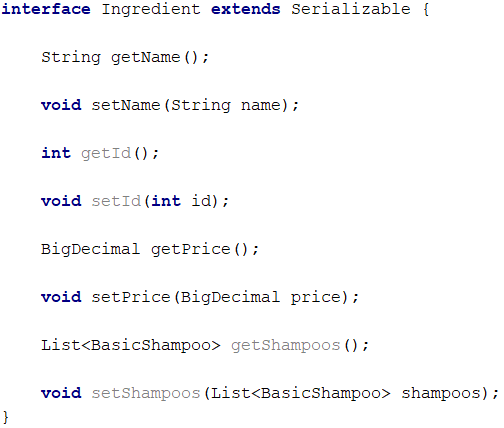
## Hierarchy Setup

Start by creating some interfaces for further implementation:

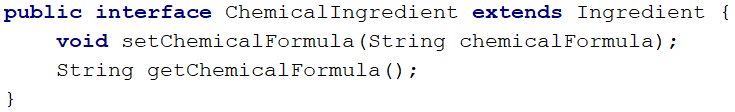
**Shampoo**:



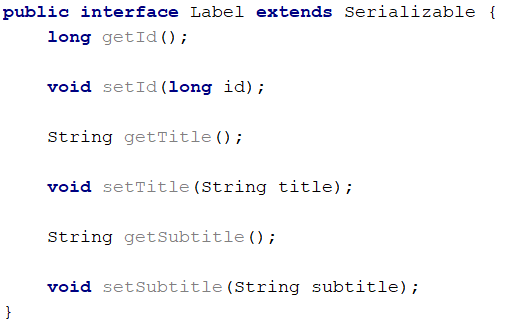
Ingredient:



ChemicalIngredient:



Label:



## Ingredients

Create root classes BasicIngredient and ChemicalIngredient, which will be extended by concrete classes later:

BasicIngredient:



All of our ingredients will be **stored in a single table** – “**ingredients**”. They will be discriminated only by their **type.**

There are two types of ingredients:

* **Basic Ingredient**. It has the following information:
  + **Id**
  + **Name**
  + **Price**
* **Chemical Ingredient** which adds additional information
  + **Chemical formula**

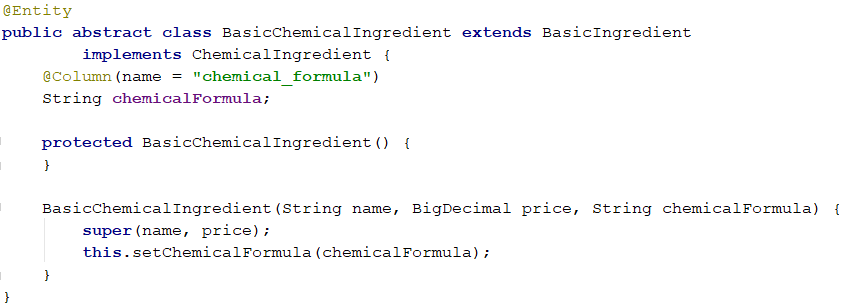
Implement have 4 types of Basic Ingredients:

* **Mint**
  + Price 3.54
* **Nettle**
  + Price 6.12
* **Strawberry**
  + Price 4.85
* **Lavender**
  + Price 2

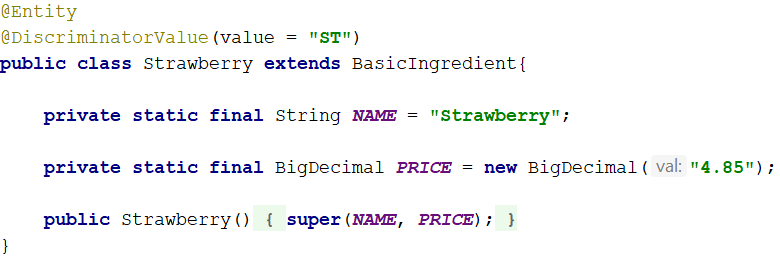
And only one Chemical Ingredient:

* **Ammonium Chloride**
  + Price 0.59
  + Formula NH4Cl

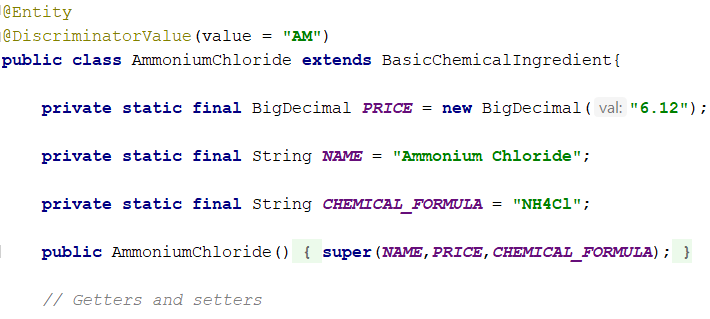
BasicChemicalIngredient:



Strawberry Ingredient:



Ammonium Chloride:

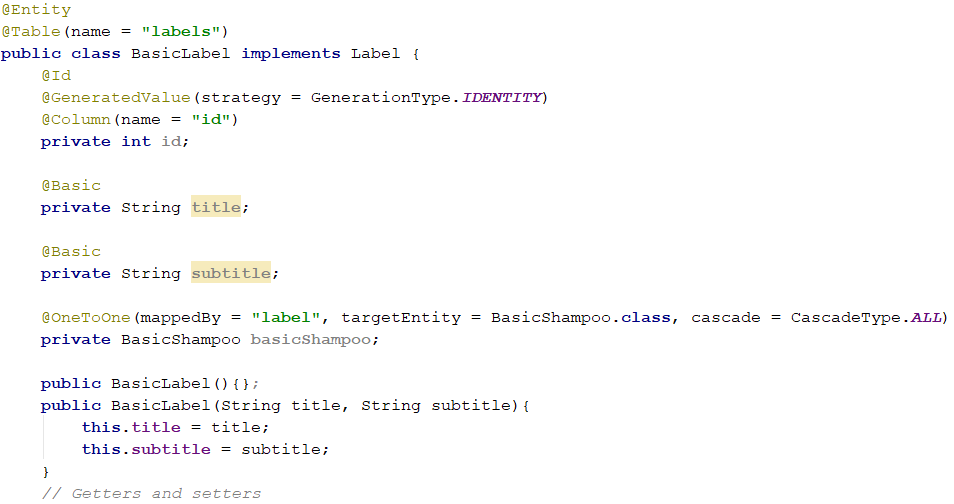


**Create the other ingredient classes analogically.**

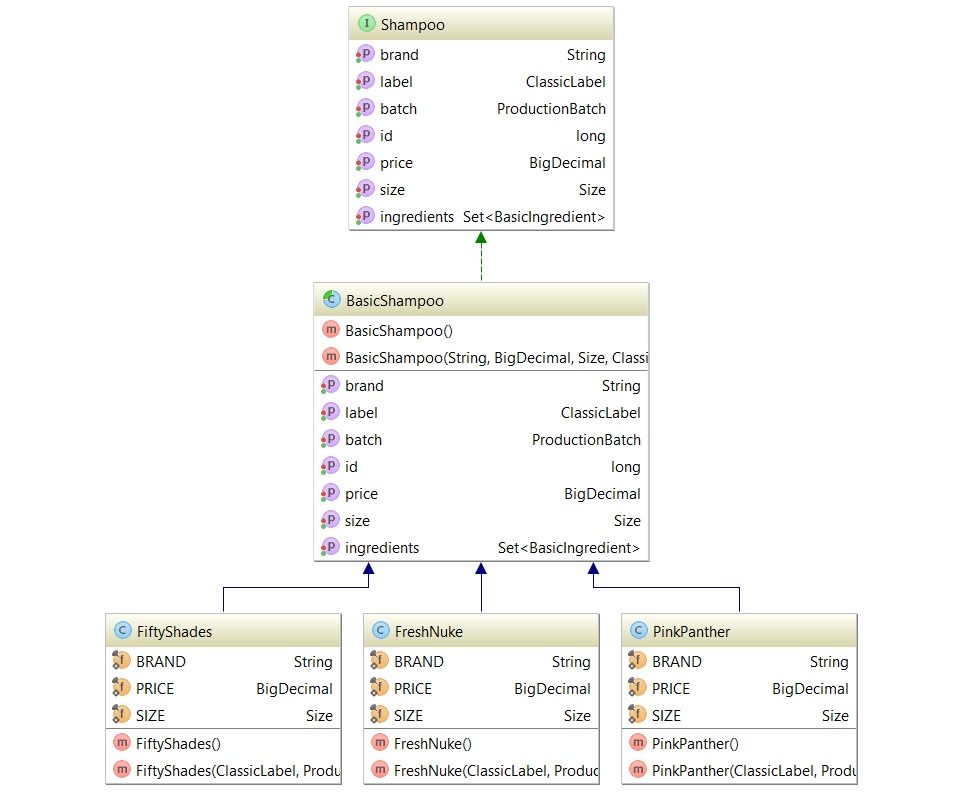
## Label

Create an abstract implementation of the Label interface you’ve created earlier.Each label will have the following fields: **id, title, subtitle.**

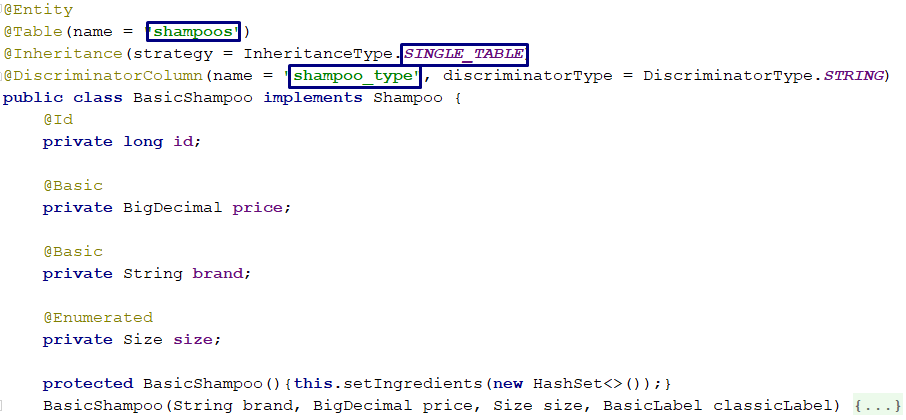
BasicLabel:



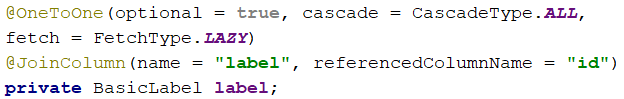
## Create Shampoos

It is required to create the following structure: 

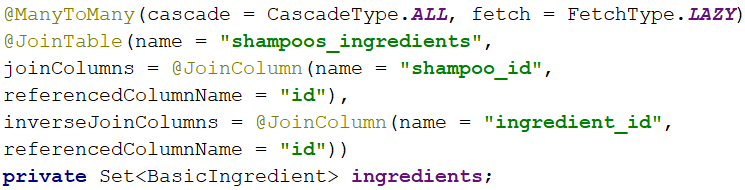
Implement the Shampoo interface you’ve created earlier, by adding a new **root** class:



Each shampoo will hold information about it’s label and ingredients. We will implement that using **table relations**. Add 2 new fields to the abstract implementation:



Every shampoo’s label will have **unidirectional** **One-to-One** relationship will a Label entity, mapped by the id **column of the label** and label **field in the shampoo implementation**.

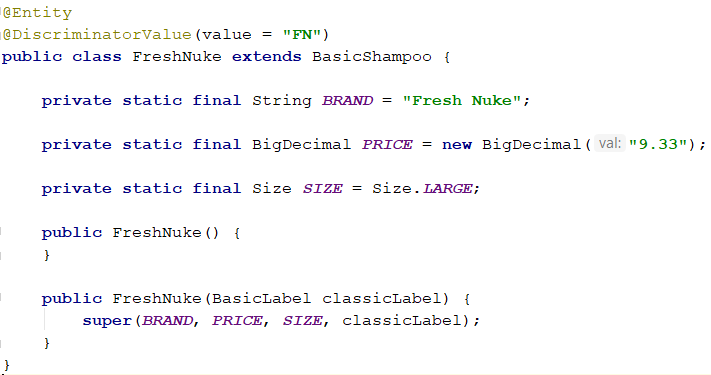


Shampoo implementations will store information about the ingredients they are made of. We will implement that by setting a bidirectional **Many-to-Many** relationship between the ingredients and each shampoo. A new annotation “JoinTable” is visible, which will create a mapping table in the database - “**shampoo\_ingredients**”. It will store the relation between each shampoo id and ingredient id.

The company produces 4 different types of Shampoos:

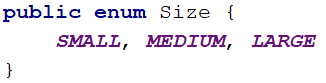
* **Fresh Nuke**
  + **Brand “Fresh Nuke”**
  + **Price 9.33**
  + **Size Big**
  + **It’s made of Mint, Nettle and Ammonium Chloride**
* **Pink Panther**
  + **Brand “Pink Panther”**
  + **Price 8.50**
  + **Size Medium**
  + **It’s made of Lavender and Nettle**
* **Fifty Shades**
  + **Brand “Fifty Shades”**
  + **Price 6.69**
  + **Size Small**
  + **It’s made of Strawberry and Nettle**

FreshNuke:



**Create the other shampoo types analogically. The inheritance should be presented in a single table.**

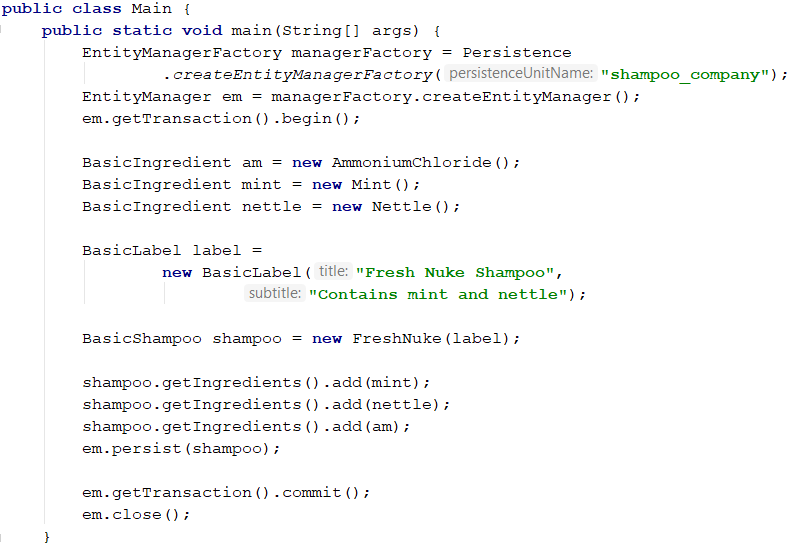
Each shampoo can be 3 sizes: SMALL, MEDIUM or BIG. Create an enumeration which will hold the size information:



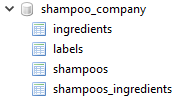
Make the field in the Shampoo class persistent by adding an @Enumerated annotation.

## Test Application

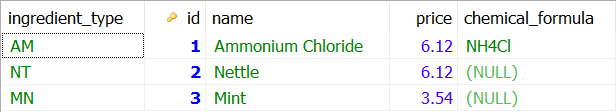
Create a **FreshNuke** shampoo. Add **nettle**, **mint** and **ammonium** **chloride** to it’s ingredients. Set up a label and persist the shampoo:



If you’ve implemented everything correctly, the following **database schema** will be created:



**Ingredients table:**



**Labels table:**



**Shampoos table:**



Shampoos and ingredients table(created via the **Many-toMany** relationship in the Shampoo class:

