**Homework 2: Designing and implementing a simple OO information system with clear separation between interfaces and implementations and DI**

The main goal of the project is to implement **Cooking Service API**allowing the ***Users*** (***HomeCooks*** and ***Administrators***) to share different ***Categories* of *Recipes*** using the system.

All Users should have following common attributes:

* ***id*** (generated automatically) - **long** number;
* ***firstName*** - **string** 2 to 15 characters long;
* ***lastName*** - **string** 2 to 15 characters long;
* ***email*** - should be ***valid*** *email* address, **unique** within the system, **cannot be changed**;
* ***username*** - **string** 2 to 15 characters long - ***word characters only***, **unique** within the system, **cannot be changed**;
* ***password*** - **string** 8 to 15 characters long, ***at least one digit***, ***one capital letter***, and ***one sign different than letter or digit***, NOT sent back to the *User* clients for security reasons;
* ***gender*** - *MALE* / *FEMALE* **enumeration**;
* ***role*** - *HOME\_COOK* or *ADMIN* **enumeration**, ***HOME\_COOK*** by default, **editable only by *Administrators***;
* ***status*** - validity status of the user account (***ACTIVE***, ***CHANGE\_PASSWORD*** - should change password on login, ***SUSPENDED***, or ***DEACTIVATED***), **visible and editable only by *Administrators***;
* - ***recipes*** - **list of all *Recipes*** added by the current ***User***;
* ***created*** (generated automatically) - time stamp of the moment the user account was created;
* ***modified*** (generated automatically) - time stamp of the moment the user account was last modified;

Additionally, the HomeCooks should have following attributes:

* ***favoriteRecipes*** - **list** of favorite ***Recipes*** chosen by the *User*;
* ***favoriteCooks*** - **list** of favorite ***HomeCooks*** (*Users*) chosen by the current *User*;

The Administrators should have in addition to the User's attributes, the following fields:

* ***categoriesModerated*** - list of all *Categories* the *Administrator* moderates *Recipes* for;

The ***HomeCooks*** should be able to manage their own **personal data** and change their **passwords**. The ***Administrators*** should be able to manage all ***User's* data**, **except their passwords**. The IDs of all entities should be generated automatically and **should NOT be changeable**;

The ***Administrators*** should be able to **add** new ***Categories*** and ***Recipes***. The ***HomeCooks*** should be able to **add new *Recipes* only** to existing *Categories*.

Each **Category** has the following structure:

* ***id*** (generated automatically) - **long** number;
* ***name*** - **string** 2 to 120 characters long;
* ***description*** (optional) - **string** 10 - 500 characters long;
* ***tags*** - string including comma separated tags, allowing to find the ***Publication*** by quick search;
* ***created*** (generated automatically) - time stamp of the moment the entity was created;
* ***modified*** (generated automatically) - time stamp of the moment the entity was last modified;

Each **Recipe** has the following structure:

* ***id*** (generated automatically) - **long** number;
* ***category*** - the *Category* the *Recipe* belongs to;
* ***title*** - **string** 2 to 120 characters long;
* ***author*** - the *User* sharing the *Recipe*;
* ***shortDescription*** - **string** 2 to 250 characters long;
* ***cookingTime*** - **integer** number in minutes;
* ***usedProducts*** - **string** 20 - 500 characters long;
* ***picture*** - of the cooked meal, **valid URL** to the picture;
* ***description*** (optional) - **string** 150 - 2500 characters long;
* ***tags*** - string including comma separated tags, allowing to find the *Recipes* by quick search;
* ***created*** (generated automatically) - time stamp of the moment the entity was created;
* ***modified*** (generated automatically) - time stamp of the moment the entity was last modified;

The ***Recipes*** should be able to find **by browsing *Categories*** and **by** ***Search*** using ***Recipe'stitle, products, short description*, and *tags***. In addition to this the ***Recipes* should be able to filter by *cooking time*** (recipes with ***cooking time*** between **minimum** and **maximum** limits).

All ***Users*** should be able to **add *Recipes*** sharing personal experience and ideas. Each ***HomeCook*** should be able to ***manage*** *(edit / delete)* **her/his own *Recipes***. The ***Administrator*** should be able **to view and *manage*** *(edit / delete)* ***Recipes*** of **all *Users***. Factory Method design pattern should be used to decouple **Services** and **Repositories** in your implementation (***Dependency Inversion principle***). *A Main class should be provided, that demonstrates the correct implementation of the above functionality.*