# ACTIVITY 5.3 – BACKEND FOR ONLINE SHOP

## INTRODUCTION

In this activity we will model the backend for an online shop or a shopping App. The data of the shop would be stored in MongoDB. The shop needs **four types of documents**: articles, comments, users and address. **The documents are related between them** as follows:

- The users have an address
- Articles have comments
- Comments are done by users

To optimize the retrieval, it has been decided that the data should be stored in **two collections**: articles and users. This would require the use of a **field in the comment to make reference to the user** author of the comment.

These are the fields for each document.

Article

- id: unique identifier of the article (use ObjectId type)
- Name: name of the article
- Price: price of the article (with decimals)
- Categories: list of words indicating different categories to which the article belongs.
- Comments: list of comments of the article

Comment

- Score: integer between 1 and 5 (control this in the setter, set 1 as min value and 5 as max value whatever the inserted value).
- User_id: ObjectId that represents the id of the user that wrote the comment
- Text: the comment content

User

- id: unique identifier of the user(use ObjectId type)
- Name: name of the user
- Email: email of the user
- Address: object of Address class representing the address of the user

Address

- Street
- Number
- City
- Country

## MODEL THE POJOS

**Write the Java classes to represent each document type.**

**Keep in mind:**

- **Article constructor** :
  - Empty
  - With arguments name, price and categories. id will be automatically set and comments will be initialized as null.
- **User constructor** :
  - Empty
  - With arguments name, email and address. id will be automatically set.

For the remaining classes you can code empty and the full arguments constructors.

## BUILD THE API

You should create a class called **DataAPI** to hide the complexity of the access to the data of the database. This would allow the developers to access the data in an easier way.

DataAPI class should have several **public and static methods to access the data**:

- `void insertArticle(Article art)`
- `void insertUser(User us)`
- `Article findArticle(ObjectId id)`
- `FindIterable<Article> findArticleByCategory(String cat):` returns all the articles that are of the category *cat*.
- `FindIterable<Article> findArticleByName(String str):` returns all the articles that contains *str* in its name.
- `FindIterable<Article> findArticleInPriceRank(double low, double high):` returns all the articles whose price is in the rank [low, high], both inclusive.
- `User findUser(ObjectId id)`
- `FindIterable<User> findUserByCountry(String country):` returns all the user who live in the *country* specified as argument.
- `FindIterable<Article> orderByPrice(FindIterable<Article> arts, boolean asc):` receives a FindIterable<Article> object and returns it ordered by price

ascending or descending as specified as argument.

- `void updateAddress (User us, Address ad)`
- `void updateEmail(User us, String email)`
- `void addComment(Article art, Comment newCom)`: **check that newCom.user_id exists as the id of a user in the Users collection**. In case of inexistence, do not add the comment but inform that the user does not exist.
- `void deleteArticle(Article art)`: delete an article.
- `void deleteUser(User us)`: delete a user **and also all the comments** whose author is the user.

Also it should have some **static attributes and methods** to manage the connection to the database:

- `MongoClient client`: **private** property to manage the MongoDB client.
- `MongoDatabase db`: **private** property to manage the MongoDB database;
- `void init()`: **public** method to initialize the MongoDB client and the database. The database is called "*act5_3*" and it should be able to manage POJOs.
- `void close()`: **public** method to close the Mongoclient.

**Test that all the methods are working properly**. For this you can create a main class that makes use of all methods and checks the result. However, this class will not be considered for assessment.