USER ADMINISTRATOR

Introduction

This document will explain the test case “**User Administration**” , where we will indicate the design of the application , and details of the implementation.

The *User Administration* application should be able via console commands.

- To create users
- To list users
- To update users
- To delete users
- Restrict manipulation operations on a user

Requirements Software

- PostgreSQL 11.x
- Java 7
- Hibernate 5

Beside the requirements I have used another 3rd party libraries like:

The *Lombok* library , It’s a library to facility the creation of the data transfer objects (DTOs). https://projectlombok.org/

With @Data anotation we have getter, setter , equals , automate the logging

The Spring framework , with SpringBoot https://spring.io/

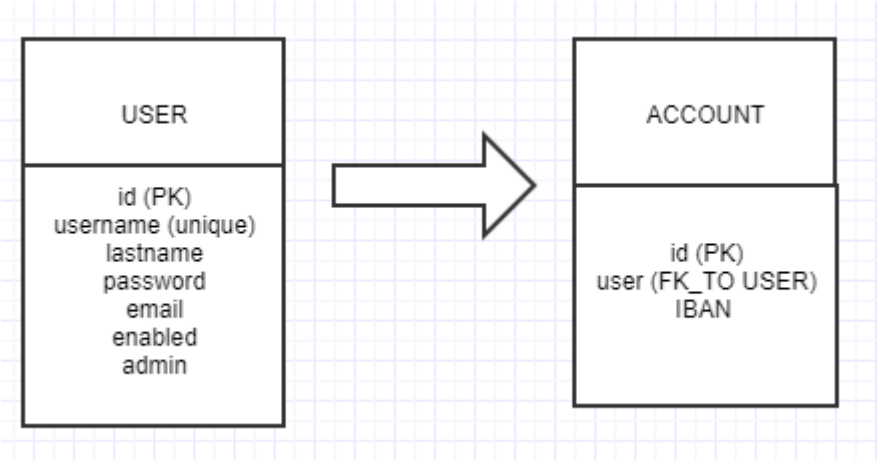
SpringSecurity used to “Restrict the operations” with the roles according the users.

Dozer, it’s a library to mapper the DTOs with the model (entities) and back

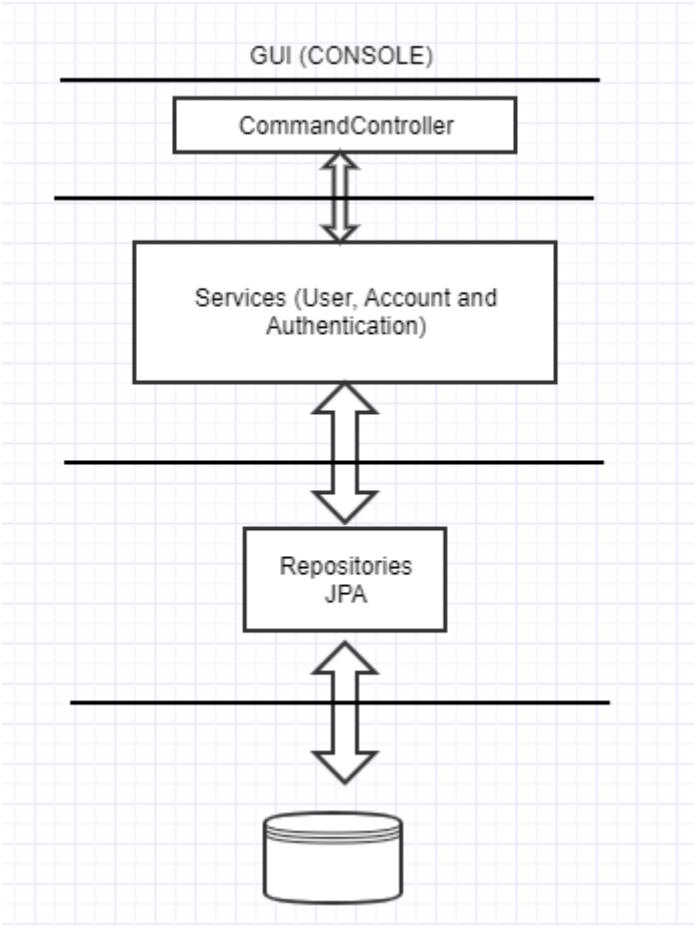
H2 It’s very useful to build test with the database operations in memory, and avoid to persits in a real database

GSON (Google JSON) , convert to JSON from DTO and convert to DTO from JSON

MAIN ENTITIES



DESIGN



RUN THE APP

The source code is on https://github.com/dgonzalezortegon/UserAdministrator

Clone the repository or Download in a folder, once you have the code, it's important to hace installed the next applications:

- Maven https://maven.apache.org/index.html
- Java 7 https://download.oracle.com/otn/java/jdk/7u80-b15/jdk-7u80-windows-i586.exe

In order to check the version, and if we have correctly installed the previous software, we can type the command "mvn -version" in a console

```
Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5; 2015-11-10T17:41:47+01:00)
Maven home: C:\tools\apache-maven-3.3.9\bin\..
Java version: 1.7.0_80, vendor: Oracle Corporation
Java home: C:\Program Files (x86)\Java\jdk1.7.0_80\jre
Default locale: es_ES, platform encoding: Cp1252
OS name: "windows 8.1", version: "6.3", arch: "x86", family: "windows"
```

- PostgreSQL 11 https://www.enterprisedb.com/
- PgAdmin (Optional) to see the data used pgAdmin 4

The PostGreSQL is running in a host:port, by default the port is 5432, but if it is another one, into the ***application-local.yml*** file must be changed in the line

url: jdbc:postgresql://localhost:5432/postgres

the ***application-local.yml*** file is located in "codeSourcefolder/src/main/resources/"

Another thing for being in mind, it's the username and password of the server

username: postgres

password: xxx

Once a time, we have installed all the software, into the /codeSourceFolder/ there is a script to run the app "run.bat" .

Or type the comand, from the script “run.bat”

mvn spring-boot:run -Dspring.profiles.active=local

The script uses ***mvn*** command, to build the application and run the console!! Reading pom.xml and completing the indicated cycle (compile and run)

The command use the ***local profile***, to read the ***application-local.yml ,***

if we want to have another profiles (preproduction , production or another configuration) we can create another ***application-profile.yml*** file changing the configuration.

USER MANUAL

console Commands

The commands are case sensitive

help *--> to visualize this help*

quit *--> exit from the console*

login username password *--> login to allow CRUD operations*

whoami *--> the current user*

create *-json {username:'userTest',password:'pass',lastname:'test',email:'test@mail.com'} -> create user*

update *userTest -json {password:'hola',lastname:'test',email:'test@mail.com'} -> Update user*

delete username ***->*** delete user with all the accounts

get username *, Get the Data of one User*

all *--> List the Data users*

newiban username iban *--> to create a new Account*

The execution of the command without arguments, show the help of the executed command

PROFILES AND OPERATIONS

The user's Operations create , update, delete and newiban (to add a new IBAN for the user) are restricted. according to the logged profile , the console allow the execution of different commands .

The two roles are ADMIN and USER, initially exist two users, one with the USER profile and another one with ADMIN profile.

The USER with the "USER PROFILE" is the username "***user***" and password "***password***"

The USER with the "ADMIN PROFILE" is the username "***admin***" and password "***password***"

These user are in memory and not persisted. (***You can use the user "admin" to start using the operations***)

OPERATIONS BY PROFILE

	create	update	delete	all	get	newiban
ADMIN	X	X	X	X	X	X
USER		X*			X*	

X* --> allow the command on the own user

The Operations: ***help , whoami, quit*** can be executed without being logged

LOGIN

The login command, support 2 arguments , username and password, both of them are case sensitive.

Examples:

login admin password

If the command is executed correctly, the console show a **true** in another case **false**

CREATE USER

The create command, support one argument, with the data of the user. The argument is a text in JSON format. Mandatory Fields “username” , “lastname” and “email”

Example

create -json {username:'userTest',password:'pass',lastname:'test',email:'test@mail.com'}

CREATE ADMIN USER

In Case to create a new Administrator, include the Attribute admin into the json

Example

create -json {username:'userTest',password:'pass',lastname:'test',email:'test@mail.com',admin:true}

UPDATE USER

The update command, support two arguments. The first argument is a username , and the second argument is a text in JSON format with the data of the user. Type the fields, that you want to update

Examples:

update userTest -json {password:'pass',lastname:'test',email:'test@mail.com'}

update userTest -json {password:'pass',email:'test@mail.com'}

//Activating user like administrator , the value ***true*** without single quotes

update userTest -json {password:'pass',email:'test@mail.com',admin:true}

DELETE USER

The delete command, support one argument, is the username of the user to be deleted

Example

delete userTest

GET USER (USER & ADMIN PROFILE)

The get command, support one argument, is the username of the user to be showed

Example

get userTest

If the user connected, is another one with the USER profile , you will receive “Access Denied” message from the console

CREATE NEW IBAN

The newiban command, support two arguments. The first argument is a username , and the second argument is the IBAN for the user.

Examples:

newiban userTest ES033456749493722

NEXT FEATURES

Feature SEND MAIL FOR THE USERS, when the user account is enabled

Feature - Password in MD5

Feature -enabled or disabled the users , like a logical delete.