**程序设计中级实践**

****

大作业

组员1

**姓名** 伊万

**学号** 6318000299

组员2

**姓名** 叶寒

**学号** 6318000426

题目: Webmarket(online store)

Our market selling “Items”. “ Items ” is things like are clothes or shoes, since it have the size parameter. The database will use column pair\_id which means a pair of things (most likely shoes). But this structure supports any type of clothing so it doesn't matter what this parameter is called.

Requirements:

1. Common page

Show list of Items, detail page, search by category. Show available Items.

1. User center

User registration, login function. Buy and reserve function. Get Wishlist, generates from reserved items. Get list of purchased Items.

1. Admin center

Admin registration, login function. Add, delete items. Add, delete items to Storage. Get list of items, list by category. Delete User. Get statistic of sold items by period.

Requirements analysis:

Following the requirements, the correct solution is to divide the service interfaces into three groups: Admin, User, and Common.

* The admin has full rights and access.
* The user can perform a limited number of actions.
* And the General interfaces only allow you to view products on the site. to purchase or reserve an item, you need to register.

We need to enter indicators that can be used to assess how successfully the project was completed.

Indicators:

* Item repository, Storage.

Save item, delete item, get all, get by column.

* Authorization

Sign-up, Sign-in for Admin and User. Interfaces limited access, only User, only Admin. Generate token.

* Common interface

Interfaces for not sign-in accesses. Show item’s catalog.

* Admin

Access to Items repository and storage. Delete Users. Get market statistic, sold or reserved items.

* User

Buy, reserve items. Get list of purchased and reserved items.

The implementation of these indicators guarantees the successful completion of the project.

Stack of technologies:

* Spring
* Spring Data
* Spring Security, JWT
* Thymeleaf (used a little bit)

Description:

This is a web service, more similar to REST services. Without front-end.

Requests: JSON

Response: JSON

Examle of Request:

Url: <http://127.0.0.1:8081/admin/auth>

Body:

{

"username" :"Ivan",

"password" : "random"

}

Response:

{

"token" :"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJLYXRlTmV3IiwiZXhwIjoxNTkxMjY1ODI5fQ.1IbhRjy4WjwQzgxoHM2aqaWHh611Z3MPVnosy3HMkn00GgNo7xxKujD0uX6sTPOM2x-vAf3eUr93NDzZjsQ8EA"

}

Structure of url’s:

Admin:

/admin /admin/storage/all

/admin/singup /admin/item/sold

/admin/auth /admin/item/sold/period

/admin/item/all /admin/users/delete

/admin/item/add

/admin/item/delete

/admin/item/category

/admin/item/gender

/admin/storage

/admin/storage/add

Admin support method:

* Add, delete, get list of items
* Add, get list, get sold list, get sold list for period for items
* Delete user

User:

/webmarket/user/signin

/webmarket/user/singup

/webmarket/user/reserve

/webmarket/user/buy

/webmarket/user/wishlist

/webmarket/user/purchased

User support methods:

* Buy, reserve item
* Get wishlist and purchased items

Public(don’t need to authenticate):

/webmarket/catalog

/webmarket/catalog/category

/webmarket/catalog/item

/webmarket/catalog/gender

/webmarket/catalog/size

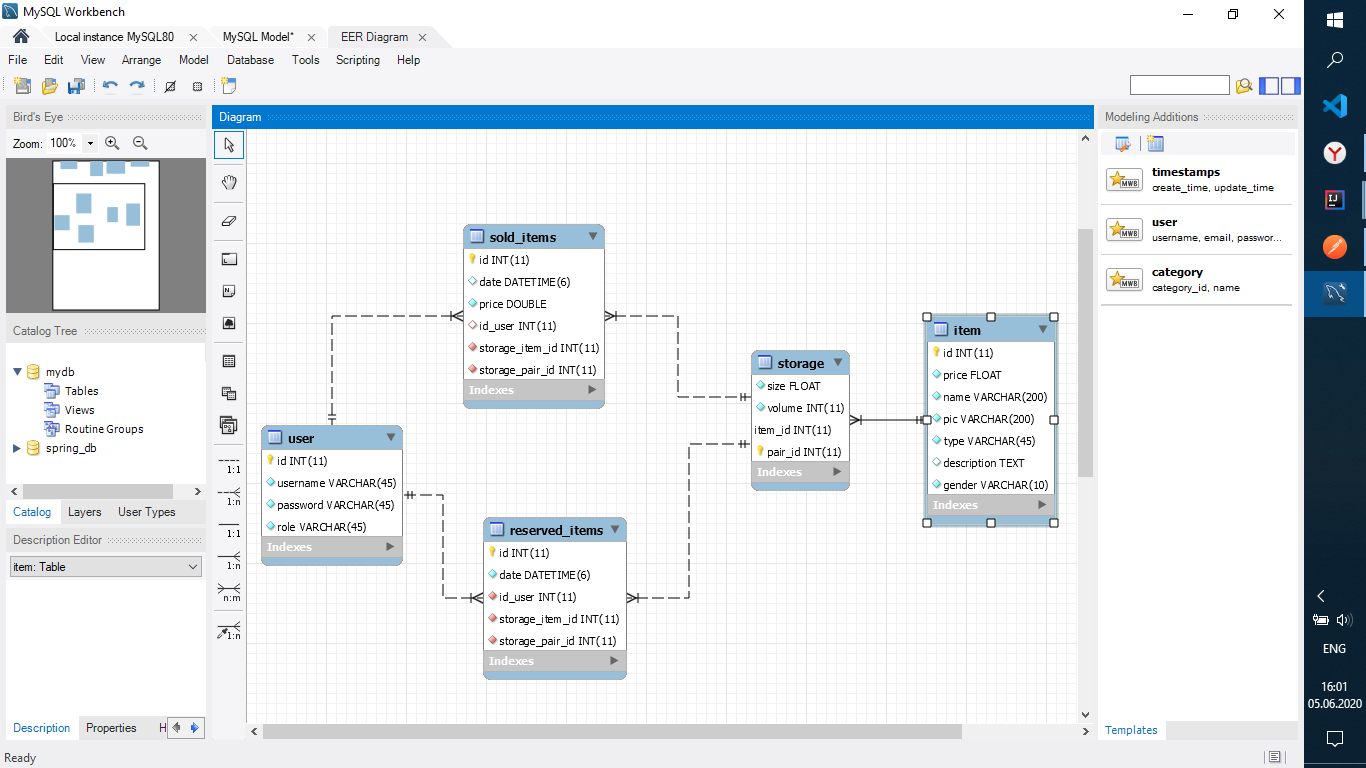
Public controller support methods:

* Get list of all item
* Get item by category, gender, size
* Get info about selected item

DB:

Example of db on git hub. File name “webmarket.sql”

Schema of DB:



Description of DB:

Item table.

Save item and main information about it.

Have ONE TO MANY relation with Storage table.

Storage table.

Save number of items(volume). It’s mean one Item have several size and we need save size of item and how may pair(pieces) of item webmarket have.

Sold\_items table:

Save history about sold items.

Reserved\_items:

Save list of reserved items

User table:

Save all Users, include Admins. There have column role, which save roles ADMIN or USER.

Security:

Using Spring Security to handel requests.Log in, sign up.

Using Jwt to generate tokens.

Personal division:

|  |  |  |
| --- | --- | --- |
| Name | Completed tasks | Source file names |
| 伊万 | 1. Implement security 2. Implement controller(API,views)   View layer   1. Create Service layer | * package webmarket.Controller;   AdminController.java, CommonController.java, UserController.java   * package webmarket.Config;   Security.java   * package webmarket.Security;   AuthenticationTokenFilter.java, AuthRequest.java, AuthResponse.java, SignUpRequest.java   * package webmarket.Util;   JwtUtil.java   * package webmarket.Service;   ItemService.java, UserDetailsService.java |
| 叶寒 | 1. Create Entities. Model layer 2. Connect DB 3. Create Service layer | * package webmarket.Entities;   Item.java, ReservedItem.java, SearchRequest.java, SoldItem.java, StorageItem.java, User.java   * package webmarket.Repository;   ItemRepository.java, ReservedItemRepository.java, SoldItemRepository.java, StorageItemRepository.java, UserRepository.java   * package webmarket.Service;   StorageItemService.java, UserService.java |