SpringBoot+Vue 的宿舍学生系统代码详解

一、项目如何设计

项目从0到1: 需求 -> 设计-> 开发-> 测试

需求:即了解项目需要什么功能,需要什么模块

设计:页面设计,画草图,数据库设计,前端后台框架架构设计,如选择 Java

开发

开发: 代码流程开发

测试:测试项目流程,测试 bug

目前项目基于 SpringBoot 开发, SpringBoot 是 SSM 框架的升级

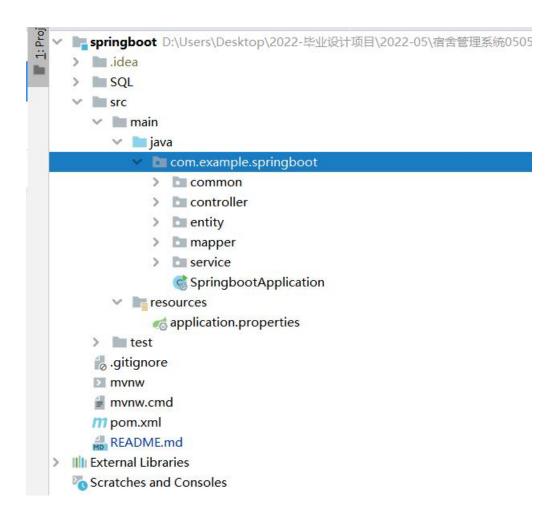
在使用传统的 Spring 去做 Java EE (Java Enterprise Edition) 开发中,大量的 XML 文件存在于项目之中,导致 JavaEE 项目变得慢慢笨重起来,繁琐的配置和整合第三方框架的配置,导致了开发和部署效率的降低。

Spring Boot 并不是用来替代 Spring 的解决方案,而是和 Spring 框架紧密结合用于提升 Spring 开发者体验的工具。同时它集成了大量常用的第三方库配置, Spring Boot 应用中这些第三方库几乎可以是零配置的开箱即用

(out-of-the-box), 大部分的 Spring Boot 应用都只需要非常少量的配置代码(基于 Java 的配置), 开发者能够更加专注于业务逻辑。

本项目是基于 Springboot + vue 前后端分离开发,技术栈特别新颖,目前企业的开发都是前后端开发,相对开发难度和成本高于传统的 Springboot 单体项目

二、项目代码结构



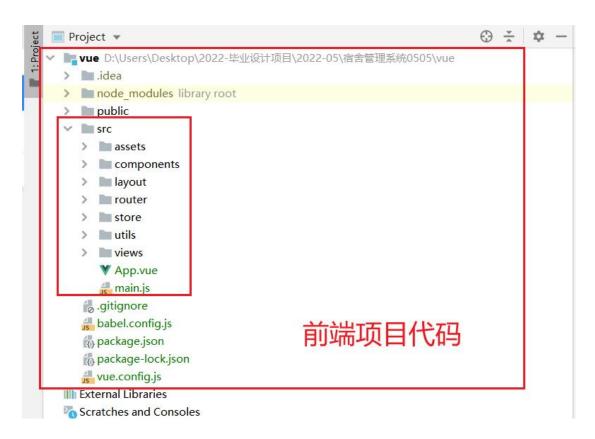
Java 相关

- 公共类 (common) 存放公共的帮助类等,如返回值公共类 Result 等等
- 表现层(controller): 负责接收客户端请求
- 实体类库(Entity): 在 Java 中,往往将其称为 Entity 实体类。数据库中用于存放数据,而我们通常选择会用一个专门的类来抽象出数据表的结构,类的属性就一对一的对应这表的属性。
- 数据访问层: (mapper 持久层)主要是对原始数据(数据库或者文本文件等存放数据的形式)的操作层,而不是指原始数据,也就是说,是对数据的操作,而不是数据库,具体为业务逻辑层或表示层提供数据服务。
- 业务逻辑层: (service)主要是针对具体的问题的操作,也可以理解成对数据层的操作,对数据业务逻辑处理,如果说数据层是积木,那逻辑层就是对这些积木的搭建。具体的讲主要负责对数据层的操作。也就是说把一些数据层的操作进行组合。

resources 资源文件:数据库的配置在里面



前端项目代码



项目运行指南:

后端运行 Springboot

https://blog.csdn.net/weixin_45393094/article/details/123674367

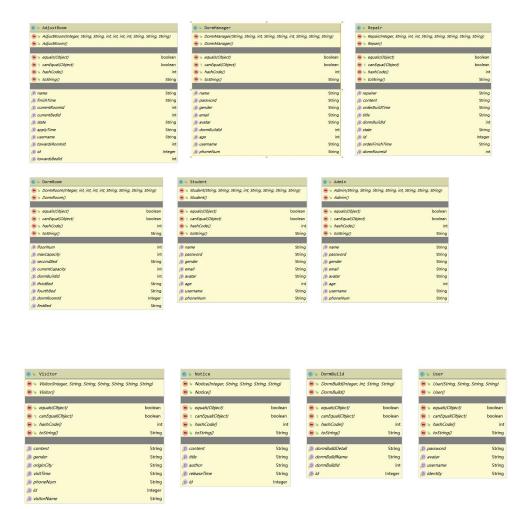
前端运行参考

https://blog.csdn.net/qq_33417321/article/details/119026234 https://blog.csdn.net/FBB360JAVA/article/details/104480578

三、时序图类图

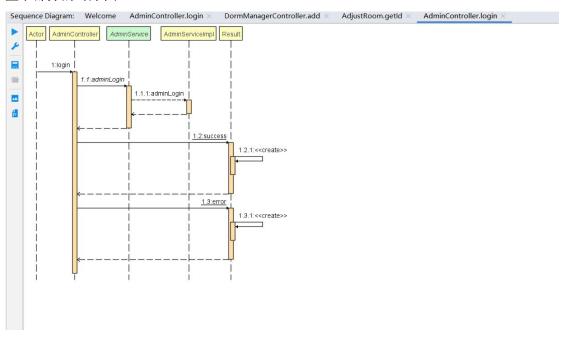
类图

就是指具体的实体类,如写文档有需要可以直接复制下面的图

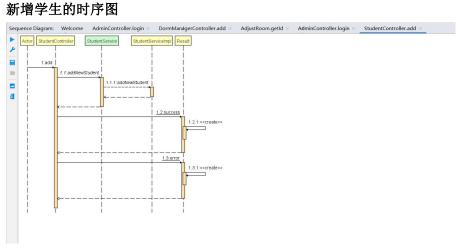


时序图

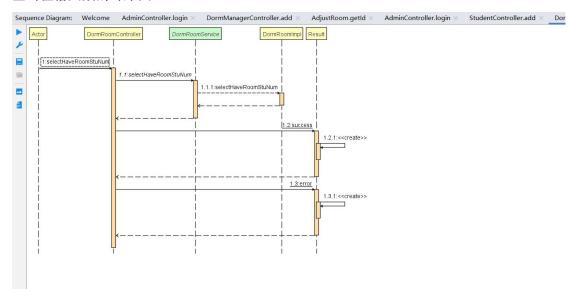
登录部分的时序图



新增学生的时序图



查询住宿人数的时序图



授人以鱼不如授人以渔

IDEA 生成时序图和类图(案例超详解)

https://blog.csdn.net/weixin_45393094/article/details/126795316

大家根据自己需要,下载完插件可以自行生成对应逻辑的时序图都有详细案例

四、具体功能对应的代码

每个模块都有对应的 controller 层代码,我也写了对应的备注,大家可以一层层的去找相关 代码

如下管理员登录相关代码

```
package com.example.springboot.controller;
  src src
       ∨ 📄 java

✓ Image: com.example.springboot

                                                                   public class AdminController {
               common
                                                                        String uid = new UID().produceUID();
                controller
                   AdjustRoomController

    AdminController

                   © DormBuildController
                                                                        private AdminService adminService;
                   © DormManagerController
                                                         21
22
23
24
25
26 @
27
28
29
30
31
32
33
34
                   © DormRoomController
                   © FileController
                                                                         * 管理品登录
                   © MainController
                   O NoticeController
                   © RepairController
© StudentController
                                                                        public Result<?> login(@RequestBody User user, HttpSession session) {
                   VisitorController
                                                                            Object o = adminService.adminLogin(user.getUsername(), user.getPassword());
               entity
                                                                                 System.out.println(o);
            > 🛅 mapper
            > service
SpringbootApplication
                                                                                 //行人session
session.setAttribute( S: "Identity", O: "admin");
session.setAttribute( S: "User", o);
   application.properties
                                                                                 return Result.success(o);
   # .gitignore

■ mvnw
                                                                                 return Result.error(code: "-1", msg: "用户名或密码错误");
   m pom.xml
                                                                         * 管理员信息更新
Illi External Libraries
```

宿管的相关代码

```
springboot D:\Users\Desktop\2022-毕业设计项目\2022-05\着 1
                                                                   package com.example.springboot.controller;
> idea
                                                                   import ...

✓ Image: src

                                                                   @RestController

✓ Imain

                                                                          estMapping("/dormManager")

√ Image: com.example.springboot

                                                                   public class DormManagerController {
            > common
                                                           16
17

    AdjustRoomController

                                                           18
19
                                                                        private DormManagerService dormManagerService;
                   AdminController
                   DormBuildControlle
                                                          20
                 © DormManagerController
© DormRoomController
                                                                          * 宿管添加
                                                                        @PostMapping("/add")
                   © FileController
                   MainController
                                                                        public Result<?> add(@RequestBody DormManager dormManager) {
                   NoticeController
                                                           25
26
27
28
29
30
31
32
33
34
                                                                            int i = dormManagerService.addNewDormManager(dormManager);
                   © RepairController
                                                                                 return Result.success();
                   © StudentController
                   © VisitorController
                                                                                 return Result.error(code: "-1", msg: "添加失败");
            > 🛅 entity
               mapper
            > m service
                SpringbootApplication
      ∨ resources
                                                                         * 宿管信息更新
             d application.properties
   > test
                                                           35
36
37
38
   🚜 .gitignore
                                                                        public Result<?> update(@RequestBody DormManager dormManager) {
   int i = dormManagerService.updateNewDormManager(dormManager);
   m pom.xml
                                                                            if (i == 1) {
```

公告相关代码

```
@RestController
       ∨ 🖿 common
                                                       15 (a)
                                                                public class NoticeController {
                CalPeopleNum
                JudgeBedName
                                                                     NoticeService noticeService;
                                                       18
19
20
21

    MyBatisPlusConfig

                © Result
                 © UID
                                                                      * 公告添加

✓ ☐ controller

                                                        22
23
24
25
26
27
28
29
30
31
32
33
34
35

    AdjustRoomController

                                                                     @PostMapping("/add")
                AdminController
                                                                     public Result<?> add(@RequestBody Notice notice) {
  int i = noticeService.addNewNotice(notice);
                 © DormBuildController
                © DormManagerController
                                                                         if (i == 1) {
                © DormRoomController
                                                                             return Result.success();
                 MainController
                                                                             return Result.error( code: "-1", msg: "添加失败");
           © NoticeController
                © RepairController
© StudentController
                © VisitorController
          > entity
                                                                       * 公告更新
          > mapper
> service
                                                       36
37
38
39
40
             SpringbootApplication
                                                                     public Result<?> update(@RequestBody Notice notice) {

✓ Imresources

                                                                         int i = noticeService.updateNewNotice(notice);
          application.properties
                                                                             return Result.success();
 .gitignore
                                                        41
42

    m∨nw

                                                                             return Result.error( code: "-1", msg: "更新失败");
 43
44
 # README.md
 External Libraries
Scratches and Consoles
```

代码基础逻辑

controller -> service -> mapper(dao)

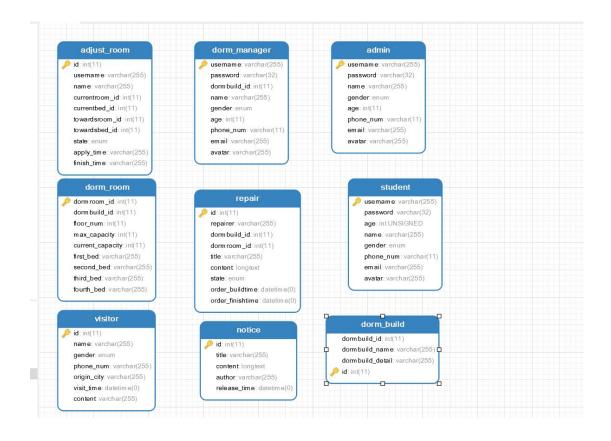
大家一层层的点击进去 即是对应的逻辑代码

关于前后端分离项目 一定要设计公共的返回值,方便前后端开发和交互,如下的截图即对 应的公共返回值



拓展:感兴趣的百度学习接口返回值设计? 为什么要设计公共的返回值?

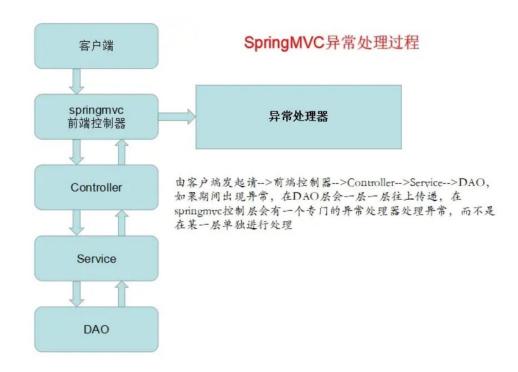
给大家截取了逆向数据库模型可以参考,即数据库的设计,具体的含义就是英文单词翻译过来,取名都比较通俗易懂



如上是比较有代表性的代码,其余的代码逻辑都是增删改查,如果不懂代码逻辑可以学习 SSM 框架,学完就懂了!

代码基础逻辑 controller -> service -> dao

如果基础薄弱的可以咨询一下身边有基础的朋友



本篇总结了项目绝大多数功能对应代码,供参考,后续不再单独提供讲解服务,祝大家 学业顺利,生活愉快!

身边有小伙伴需要毕业设计可以推荐,作为回报,我可以发红包或者指导一次毕业设计毕业生简历无偿帮助辅导修改

基于 SpringBoot+Vue 的学生宿舍管理系统(Java 毕业设计)

https://blessing.blog.csdn.net/article/details/125019849

本篇原创: 辰兮 2022.09.10