《J2EE 互联网程序设计》项目开发文档

第 34 组 项目 名称：swust BBS 信息交流平台

学号	姓名	专业班级	组长
5120165564	胡耀	卓软 1601	√
5120162272	谢欣玉	卓软 1601	
5120162255	朱洪成	卓软 1601	
5120162126	周隆放	卓软 1601	

文档评分表

#	评分点	描述	满分	得分	小计	评阅人
1	项目概述	项目概述和项目功能描述、数据准备	5			
2	工程构建	项目构建等表述	5			
3	HTTP Request	Contoller RESTful 部分	5			
4	数据持久化	Spring DataJPA 部分	5			
5	复杂查询设计	Spring Data JPA 复杂查询部分	5			
6	前端设计思路	前端部分功能关键思路和关键代码	5			

2019 年 6 月 17 日

计算机科学与技术学院

一、项目背景描述：

目前，有大约 50%的西科大在校大学生喜欢使用跳蚤街来帮助自己获得信息，资源。但 80%的大学生认为他们的需求不能在自己预想时间内解决。
其大部分原因在于自己发布的信息不能被信息发布员及时的发布出去，并且跳蚤街群人数有限而发布信息量大，导致自己发布的信息很快被其他信息淹没，并且由于人数限制，能帮助的人少。 据调查，有 68.2%的人有遇到过花费很大精力去寻找自己想要的人力或者物品资源.有 80%的人有想过利用自己的空余时间去做一些事.因此有 93.2%的人希望能有一款软件能解决发布信息迟，查看发布信息的人数有限的问题。

立项：我们团队想做一款西科互助的网站满足这些日常生活中具有强烈需求的大学生，来实现能够不依靠信息发布员，可以自己发布信息或者通过自己查找信息帮助那些有需求的人，从而获取报酬，还能够十分方便的分类查找各种信息。

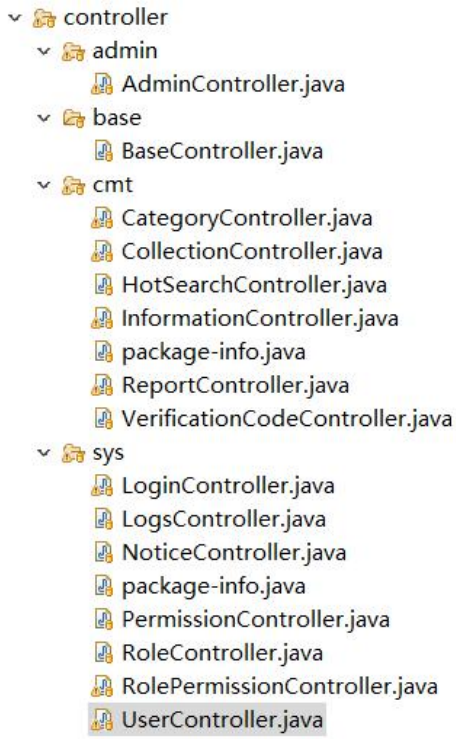
二、项目及开发环境的构建

开发环境： node.js (v10)
页面框架： Vue.js，springboot+mybatis
后台开发框架：spring boot + mybatis
组件库： ElementUI
网络请求： axios
数据库：mysql
开发工具: sts(spring boot suite)+navicate premium

使用 vue-cli 快速搭建结构，并在项目中添加 elementUI。
安装依赖 axios 。 配置请求转发代理，因为前后端有跨域，利用 webpack 的能力将前端以 ’api’ 开头的请求都转发到后端的 localhost:8085 接口。
配置 vue-router，使得前端页面可以多页面中展示。

三、HTTP Request 中 RESTful API 及 Controller 设计

controler 总共分成如下的的结构：



Cmt 的功能都从 BaseController 这样的基类中继承而来。

1.cmt 中 CategoryContrller 是实现分类功能的 Controller

```java
package com.team2.controller.cmt;
import javax.validation.Valid;
@Api(tags = "分类")
@RestController
@Validated
@RequestMapping("/category")
public class CategoryController extends BaseController {

    @Autowired
    public CategoryService categoryService;

    //@PreAuthorize("hasAuthority('ADMIN')")
    @ApiOperation(value="删除记录")
    @DeleteMapping(value="/del")
    public ResultResponse<Long> delById(@ApiParam(required=true,value="查询编号")
        @RequestBody BaseRecord recode) {
        Long count =categoryService.delById(recode.getId());
        return new ResultResponse<Long>(count);
    }

    //@PreAuthorize("hasAuthority('ADMIN')")
    @ApiOperation(value="新增记录")
    @PostMapping(value="/add")
    public ResultResponse<Category> save(@ApiParam(required = true, value = "添加Category")@Vali
        @RequestBody Category record) {
        categoryService.save(record);
        return new ResultResponse<Category>(0,"ok", record);
    }

```

定义了删除、新增、查询、修改、查看详情的功能函数。

2.cmt 中 CollectionContrller 是实现收藏功能的 Controller

```java
package com.team2.controller.cmt;

import java.util.List;

@Api(tags = "收藏")
@RestController
@Validated
@RequestMapping("/collection")
public class CollectionController extends BaseController {

    @Autowired
    public CollectionService collectionService;

    @ApiOperation(value = "删除记录")
    @DeleteMapping(value = "/del")
    public ResultResponse<Long> delById(@ApiParam(required = true, value = "查询编号") @RequestBody B
        Long count = collectionService.delById(recode.getId());
        return new ResultResponse<Long>(count);
    }

    @ApiOperation(value = "新增记录")
    @PostMapping(value = "/add")
    public ResultResponse<Collection> save(
            @ApiParam(required = true, value = "添加Collection") @Valid @RequestBody Collection recor
        Long l = collectionService.save(record);
        if (l > 0L) {
            return new ResultResponse<Collection>(0, "ok", record);
        } else {
            return new ResultResponse<Collection>(-1, "该条信息已收藏，不能收藏", record);
        }
    }
```

该类 CollectionController 定义了删除、新增、查询、修改、查看详情、获取我的信息的功能函数。

3.cmt 中 InformationContrller 是实现查看信息的功能的 Controller

```
import java.io.File;
@Api(tags = "信息")
@RestController
@Validated
@RequestMapping("/information")
public class InformationController extends BaseController {

    @Autowired
    public InformationService informationService;

    @Value("${files.picpath}")
    private String filesPath;

    @ApiOperation(value="删除记录")
    @DeleteMapping(value="/del")
    public ResultResponse<Long> delById(@ApiParam(required=true,value="查询编号")
        @RequestBody BaseRecord recode) {
        Long count =informationService.delById(recode.getId());
        return new ResultResponse<Long>(count);
    }

    @ApiOperation(value="新增记录")
    @PostMapping(value="/add")
    public ResultResponse<Information> save(@ApiParam(required = true, value = "添加Information")@Valid
        @RequestBody Information record) {
        Long long1=informationService.save(record);
        return new ResultResponse<Information>(0,"ok", record);
    }

    @ApiOperation(value = "根据ID查询记录")
    @PutMapping(value = "/get")
```

该类中定义了删除、新增、查询、修改、修改、获取我的发布需求、查看详情、信息分类、查看统计数据、上传图片的功能函数。

4.Sys 模块中 Login 是实现登陆的功能

```
import java.util.HashMap;

@Api(tags = "登录")
@RestController
@Validated
@RequestMapping("/")
public class LoginController {
    @Autowired
    private UserService userService;
    @Autowired
    private TokenService tokenService;
    @Autowired
    private AuthenticationManager authenticationManager;
    @Autowired
    private IMService imService;
    private final Logger logger = LoggerFactory.getLogger(LoginController.class);

    @ApiOperation(value = "登录")
    @PostMapping(value = "/login")
    public void login(@RequestBody LoginInfo loginInfo, HttpServletRequest request, HttpServl
        try {
            UsernamePasswordAuthenticationToken authRequest = new UsernamePasswordAuthenticat
                    loginInfo.getPassword());
            Authentication authentication = authenticationManager.authenticate(authRequest);
            SecurityContextHolder.getContext().setAuthentication(authentication);
            HttpSession session = request.getSession();
            session.setAttribute("SPRING_SECURITY_CONTEXT", SecurityContextHolder.getContext(

            LoginUser loginUser = (LoginUser) authentication.getPrincipal();
            Token token = tokenService.saveToken(loginUser);
            //更新IM的token.
            imService.UpdateAccount(String.valueOf(loginUser.getId()),token.getToken());
```

5.Sys 模块中 UserControllser 是实现用户管理的功能

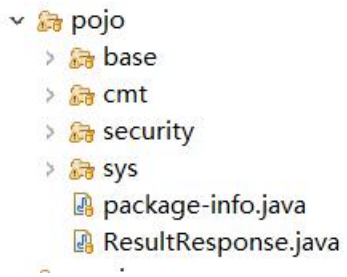
```java
import java.io.File;

@Api(tags = "用户")
@RestController
@Validated
@RequestMapping("/user")
public class UserController extends BaseController {

    @Autowired
    public UserService userService;
    @Autowired
    public UserDao userDao;

    @Autowired
    public IMService imService;
    @Value("${files.path}")
    private String filesPath;

    @ApiOperation(value = "删除记录")
    @DeleteMapping(value = "/del")
    public ResultResponse<Long> delById(@ApiParam(required = true, value = "查询编号") @Requ
        Long count = userService.delById(recode.getId());
        return new ResultResponse<Long>(count);
    }

    @ApiOperation(value = "新增记录")
    @PostMapping(value = "/add")
    public ResultResponse<User> save(@ApiParam(required = true, value = "添加User") @Valid @
        Long accid = userService.save(record);
        int status=imService.CreateAccount(accid.toString(),record.getNickname());
        return new ResultResponse<User>(status, "ok", record);
    }

    @ApiOperation(value = "根据ID查询记录")
    @PutMapping(value = "/get")
    public ResultResponse<User> getById() {
        User record = userService.getUser();
        return new ResultResponse<User>(record);
```

该类中定义了用户的增、删、改、查记录的功能，查看用户信息、修改用户信息、上传用户头像、修改密码的功能。

四、数据持久化中 Entity 及 Repository 的设计

Entity 设计：

Entity 的结构如下：



cmt 中分成 Category、Collection、Information 实体

1.Category 实体：

```java
package com.team2.pojo.cmt;

import java.io.Serializable;
@JsonInclude(Include.NON_NULL)

/**Category*/
public class Category implements Serializable {

    @ApiModelProperty(value = "")
    private Long id;

    @ApiModelProperty(value = "分类名称(字符长度为1-255)")
    @Size(min=1, max=255,message="分类名称字符长度为1-255")
    private String name;

    @ApiModelProperty(value = "类型0：需求1：资源(字符长度为1-1)")
    @Size(min=1, max=1,message="类型0：需求1：资源字符长度为1-1")
    private String type;


    private static final long serialVersionUID = 1L;

    public Long getId () {
        return id;
    }

    public void setId (Long id) {
        this.id= id ;
    }

    public String getName () {
         return name;
    }

```

设置 id,name,type 的属性，以及设置、获得这些属性的接口。

2.Collection 实体：

```java
package com.team2.pojo.cmt;

import java.io.Serializable;
@JsonInclude(Include.NON_NULL)

/**Collection*/
public class Collection implements Serializable {

    @ApiModelProperty(value = "")
    private Long id;

    @ApiModelProperty(value = "信息id")
    private Long informationId;

    @ApiModelProperty(value = "用户id")
    private Long userId;

    @ApiModelProperty(value = "创建时间")
    private Timestamp createTime;


    private static final long serialVersionUID = 1L;

    public Long getId () {
        return id;
    }

    public void setId (Long id) {
        this.id= id ;
    }

    public Long getInformationId () {
```

设置了 id，informationId,userId,createTime 的属性，以及对这些属性进行修改、获取的方法。

3.Information 的实体设计

```
    @ApiModelProperty(value = "")
    private Long id;

    @ApiModelProperty(value = "标题(字符长度为1-255)")
    @Size(min=1, max=255,message="标题字符长度为1-255")
    private String title;

    @ApiModelProperty(value = "正文内容(字符长度为1-255)")
    @Size(min=1, max=255,message="正文内容字符长度为1-255")
    private String content;

    @ApiModelProperty(value = "分类名称")
    private String name;

    @ApiModelProperty(value = "图片(字符长度为1-255)")
    @Size(min=1, max=255,message="图片字符长度为1-255")
    private String picture;

    @ApiModelProperty(value = "信息状态0：未交易1：已交易2：涉嫌违规(字符长度为1-1)")
    @Size(min=1, max=1,message="信息状态0：未交易1：已交易2：涉嫌违规字符长度为1-1")
    private String status;

    @ApiModelProperty(value = "信息类型0：需求1：资源(字符长度为1-1)")
    @Size(min=1, max=1,message="信息类型0：需求1：资源字符长度为1-1")
    private String type;

    @ApiModelProperty(value = "分类id")
    private Long categoryId;

    @ApiModelProperty(value = "用户id")
    private Long userId;
```

```
@ApiModelProperty(value = "用户id")
private Long userId;

@ApiModelProperty(value = "用户昵称")
private String userNickName;

@ApiModelProperty(value = "用户头像URL")
private String userHeadUrl;

@ApiModelProperty(value = "创建时间")
private Timestamp createTime;

@ApiModelProperty(value = "修改时间")
private Timestamp updateTime;

@JsonFormat(locale="zh", timezone="GMT+8", pattern="yyyy-MM-dd")
@ApiModelProperty(value = "失效时间")
private Date endTime;


private static final long serialVersionUID = 1L;
```

设置了标题、正文内容、分类名称、图片、信息状态、分类、用户 id、用户昵称、用户头像、创建时间、修改时间、失效时间。以及对这些属性进行访问、修改的方法。

4.LoginUser 实体设计：

```java
package com.team2.pojo.sys;

import java.util.ArrayList;

public class LoginUser extends User implements UserDetails{

    private List<Permission> permissions;

    private String token;
    /** 登陆时间戳（毫秒） */
    private Long loginTime;
    /** 过期时间戳 */
    private Long expireTime;

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
//      List<GrantedAuthority> auths = new ArrayList<>();
//       if (this!=null && this.getRoleId()==1)
//           auths.add(new SimpleGrantedAuthority("ADMIN"));
//       else if(this!=null && this.getRoleId()==2)
//           auths.add(new SimpleGrantedAuthority("USER"));
//       return auths;
        return null;
    }

    @Override
    public boolean isAccountNonExpired() {
        // TODO Auto-generated method stub
        return true;
    }

    @Override
```

设计了 登陆时间、过期时间的属性，定义了对属性的修改和访问的方法。

5.User 实体设计：

```java
/**User*/
public class User implements Serializable {

    @ApiModelProperty(value = "")
    private Long id;

    @ApiModelProperty(value = "用户名(手机号)")
    private String username;

    @ApiModelProperty(value = "昵称(字符长度为1-255)")
    @Size(min=1, max=255,message="昵称字符长度为1-255")
    @Excel(name="昵称",orderNum="1")
    private String nickname;

    @ApiModelProperty(value = "手机(字符长度为1-255)")
    @Size(min=1, max=255,message="手机字符长度为1-255")
    @Excel(name="手机",orderNum="2")
    private String telephone;

    @ApiModelProperty(value = "密码(字符长度为1-255)")
    //@Size(min=1, max=255,message="密码字符长度为1-255")
    @Excel(name="密码",orderNum="3")
    private String password;

    @ApiModelProperty(value = "头像(字符长度为1-255)")
    @Size(min=1, max=255,message="头像字符长度为1-255")
    @Excel(name="头像",orderNum="4")
```

```
    private String headImgUrl;

    @ApiModelProperty(value = "生日")
    private Date birthday;

    @ApiModelProperty(value = "性别(字符长度为1-1)")
    @Size(min=1, max=1,message="性别字符长度为1-1")
    @Excel(name="性别",orderNum="6")
    private String sex;

    @ApiModelProperty(value = "状态：不可用 0;可用 1;锁定2;")
    private String status;

    @ApiModelProperty(value = "创建时间")
    private Timestamp createTime;

    @ApiModelProperty(value = "修改时间")
    private Timestamp updateTime;

    @ApiModelProperty(value = "个性签名(字符长度为1-255)")
    @Size(min=1, max=255,message="个性签名字符长度为1-255")
    @Excel(name="个性签名",orderNum="10")
    private String signature;

    @ApiModelProperty(value = "家乡(字符长度为1-255)")
    @Size(min=1, max=255,message="家乡字符长度为1-255")
    @Excel(name="家乡",orderNum="11")
    private String hometown;

    @ApiModelProperty(value = "角色id")
    private Long roleId;
```

设置了 id，username，telephone，password，头像，生日，性别，状态，修改时间，个性签名、家乡、角色等属性，并设置对对这些属性进行访问、修改的的方法。

Repository 设计：

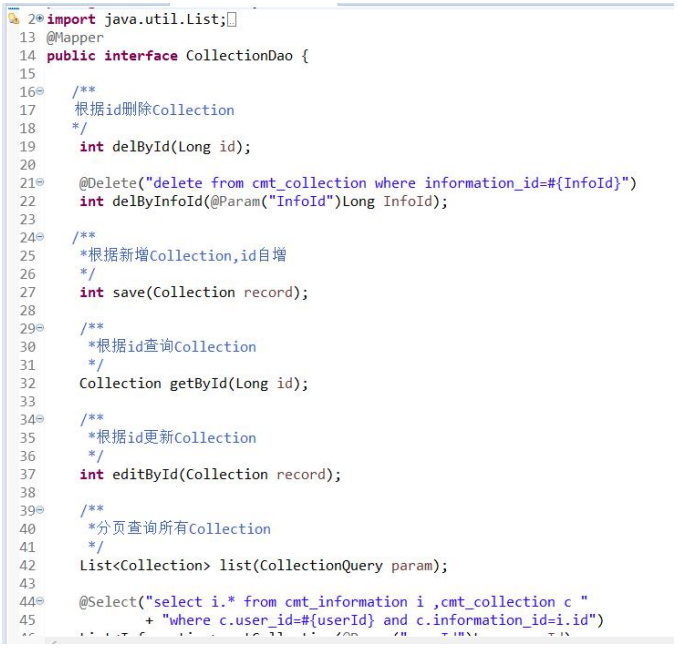
1.Category 的 Repository：

```
package com.team2.dao.cmt;
import java.util.List;
@Mapper
public interface CategoryDao {

    /**
    根据id删除Category
    */
     int delById(Long id);

    /**
     *根据新增Category,id自增
     */
     int save(Category record);

     /**
      *根据id查询Category
      */
     Category getById(Long id);

     /**
      *根据id更新Category
      */
     int editById(Category record);

     /**
      *分页查询所有Category
      */
     List<Category> list(CategoryQuery param);


}
```

设置了根据 id 删除 Category、根据 id 新增 Category、根据 id 查询 Category、

根据 id 更新 Category，分页查询所有 Category。

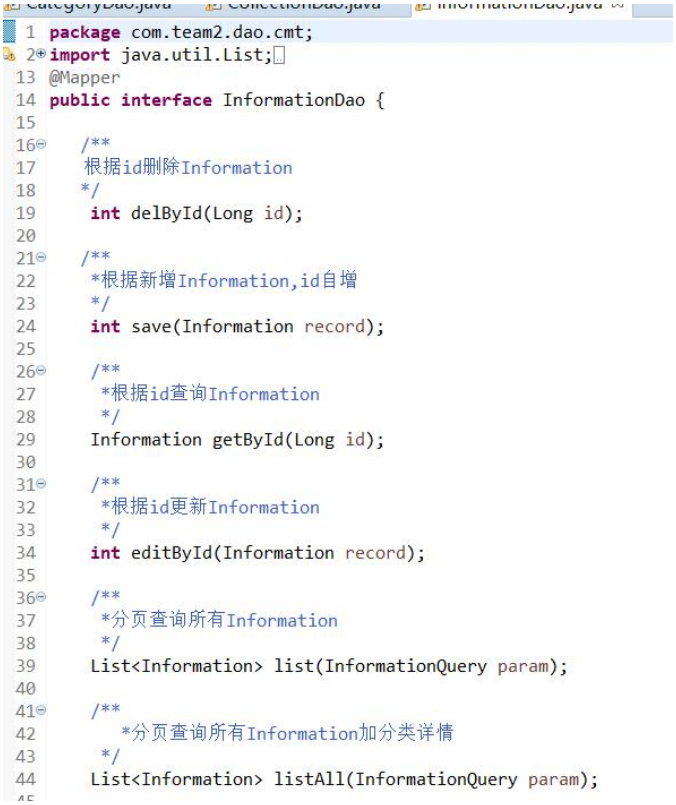
2.Collecton 的 Repository:



```
import java.util.List;
@Mapper
public interface CollectionDao {

    /**
    根据id删除Collection
    */
     int delById(Long id);

     @Delete("delete from cmt_collection where information_id=#{InfoId}")
     int delByInfoId(@Param("InfoId")Long InfoId);

    /**
     *根据新增Collection,id自增
     */
     int save(Collection record);

     /**
      *根据id查询Collection
      */
     Collection getById(Long id);

     /**
      *根据id更新Collection
      */
     int editById(Collection record);

     /**
      *分页查询所有Collection
      */
     List<Collection> list(CollectionQuery param);

     @Select("select i.* from cmt_information i ,cmt_collection c "
             + "where c.user_id=#{userId} and c.information_id=i.id")
```

设置了删除 Collection、新增 Colletion、查询 Colletion、更新 Collection、分页查询 Collection。

3.Infomation 的 Repository 设计：



```
package com.team2.dao.cmt;
import java.util.List;
@Mapper
public interface InformationDao {

    /**
    根据id删除Information
    */
     int delById(Long id);

     /**
      *根据新增Information,id自增
      */
     int save(Information record);

     /**
      *根据id查询Information
      */
     Information getById(Long id);

     /**
      *根据id更新Information
      */
     int editById(Information record);

     /**
      *分页查询所有Information
      */
     List<Information> list(InformationQuery param);

     /**
        *分页查询所有Information加分类详情
      */
     List<Information> listAll(InformationQuery param);
```

设置了删除、新增、查询、更新、分页查询 Information。

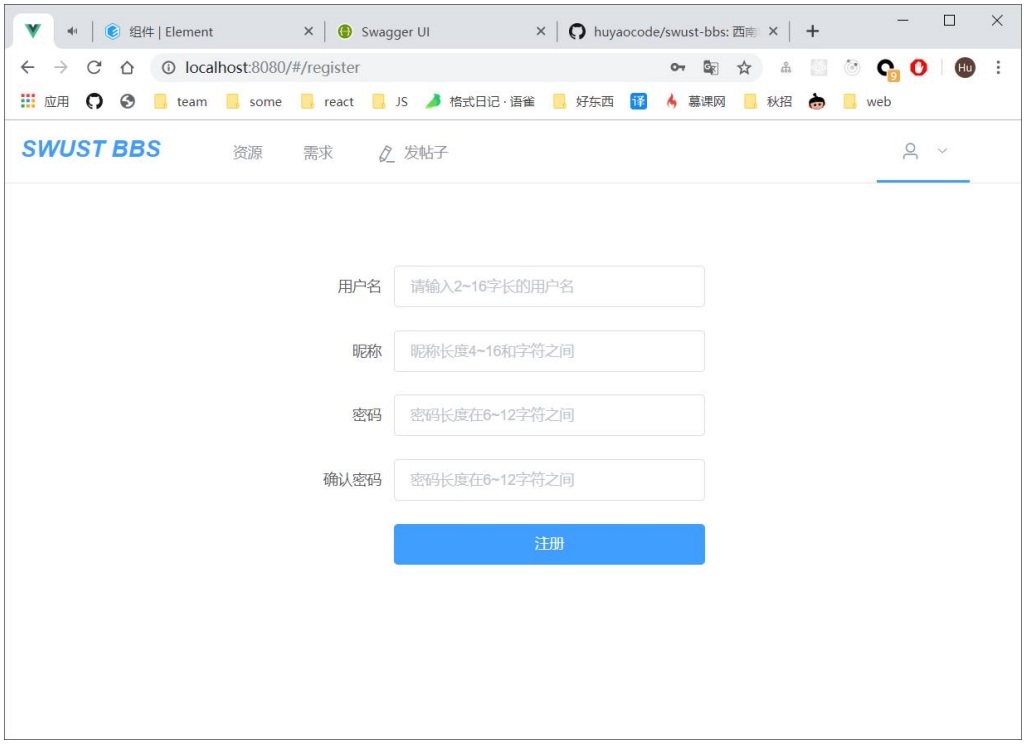
4.User 的 Reposity 设计：

```
package com.team2.dao.sys;
import java.util.List;
@Mapper
public interface UserDao {

   /**
    根据id删除User
   */
    int delById(Long id);

   /**
    *根据新增User,id自增
    */
    int save(User record);


    @Update("update sys_user set password=#{password} where id=#{id}")
    int editPassword(@Param("password")String password,@Param("id")Long id);

    /**
     *根据id查询User
     */
    User getById(Long id);

    /**
     *根据id更新User
     */
    int editById(User record);

    /**
     *分页查询所有User
     */
    List<User> list(UserQuery param);

    @Select("select * from sys_user where name username #{username}")
```

设置了删除、新增、查询、更新、分页查询的所有 User。

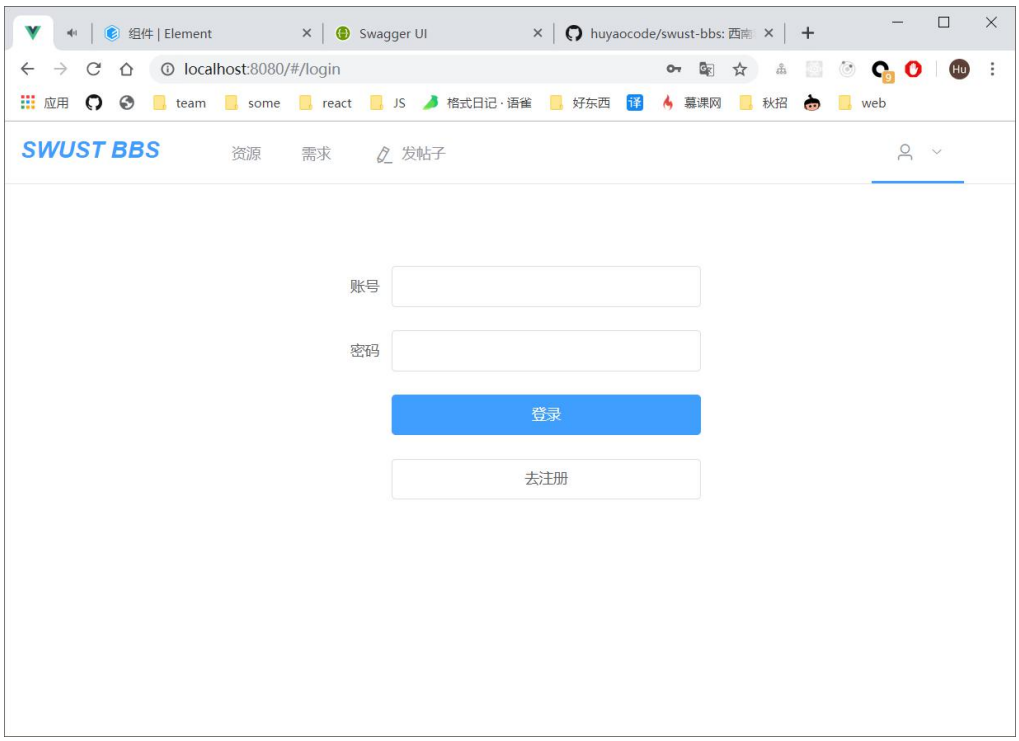
五、前端页面布局与数据交互设计

1.注册界面：



需要输入用户名、昵称、密码以及确认密码

2. 登录界面：

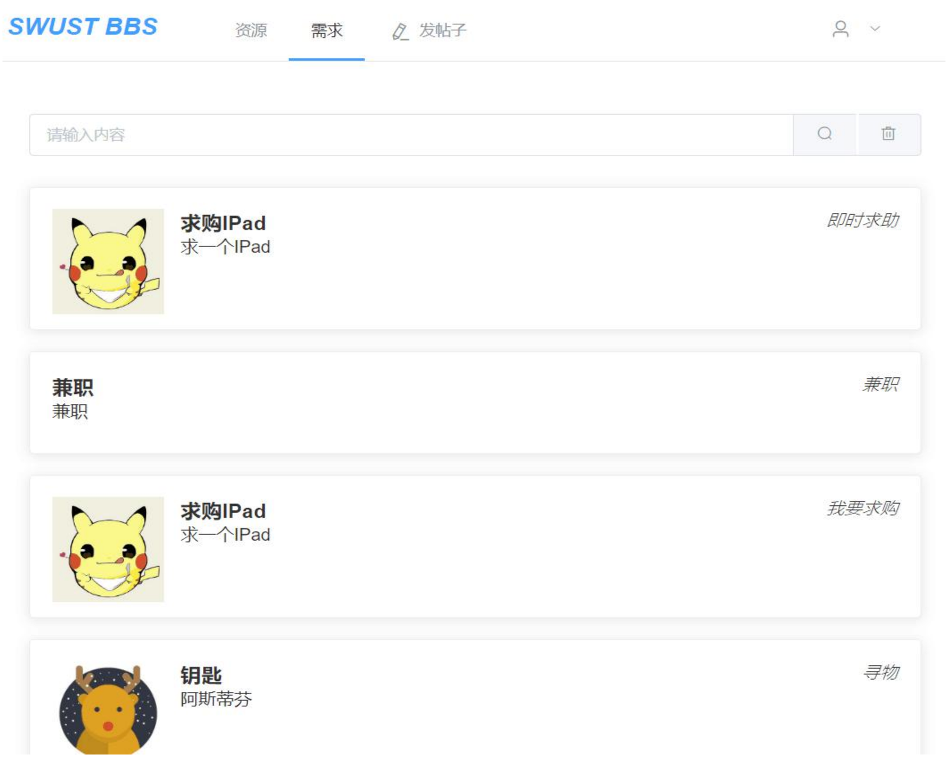


输入账号昵称即可

3. 查看资源信息：

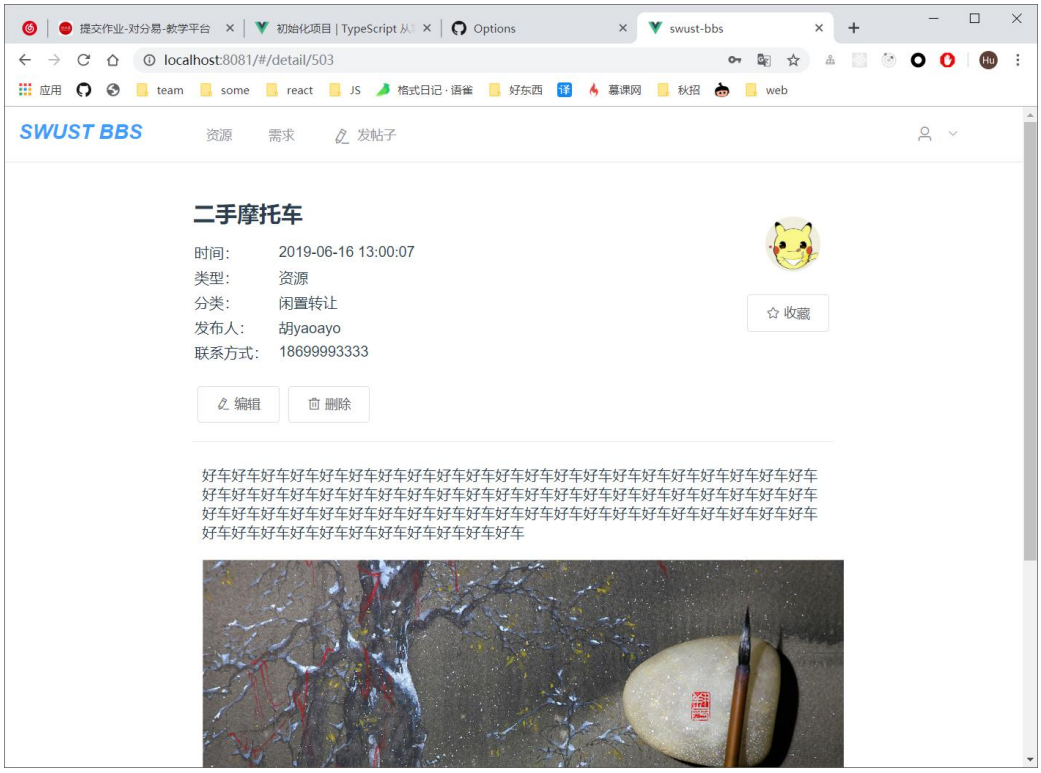
可以一个信息的缩略图，包括、图片、名称、简介。

4. 需求列表：



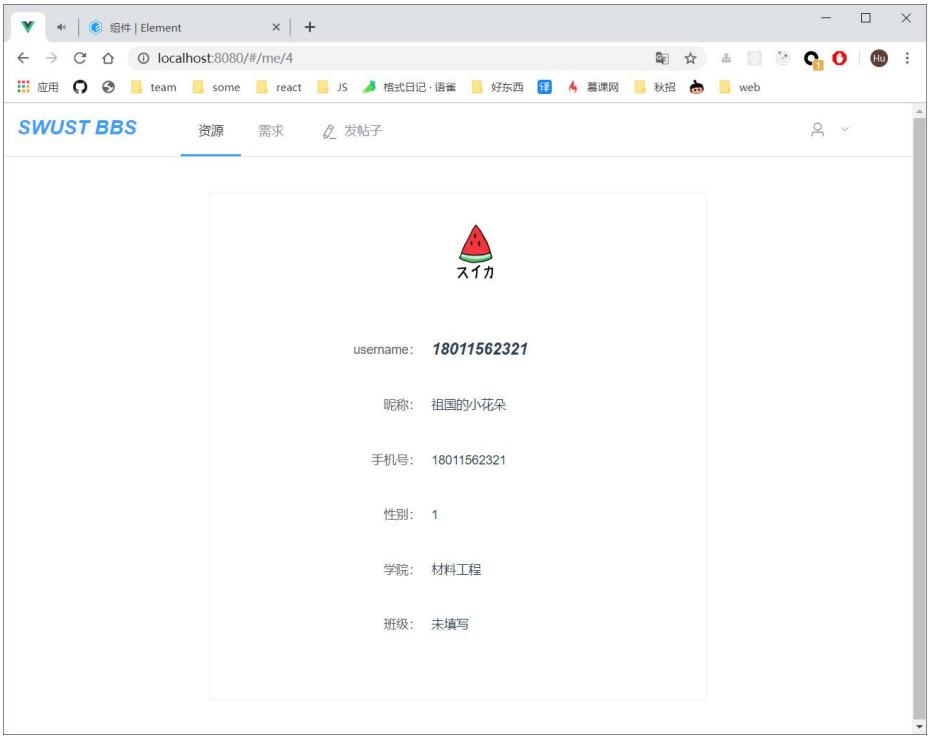
可以展现需求的缩略图，包括头像、类型、信息名称、用户描述等。

5. 信息详情界面：



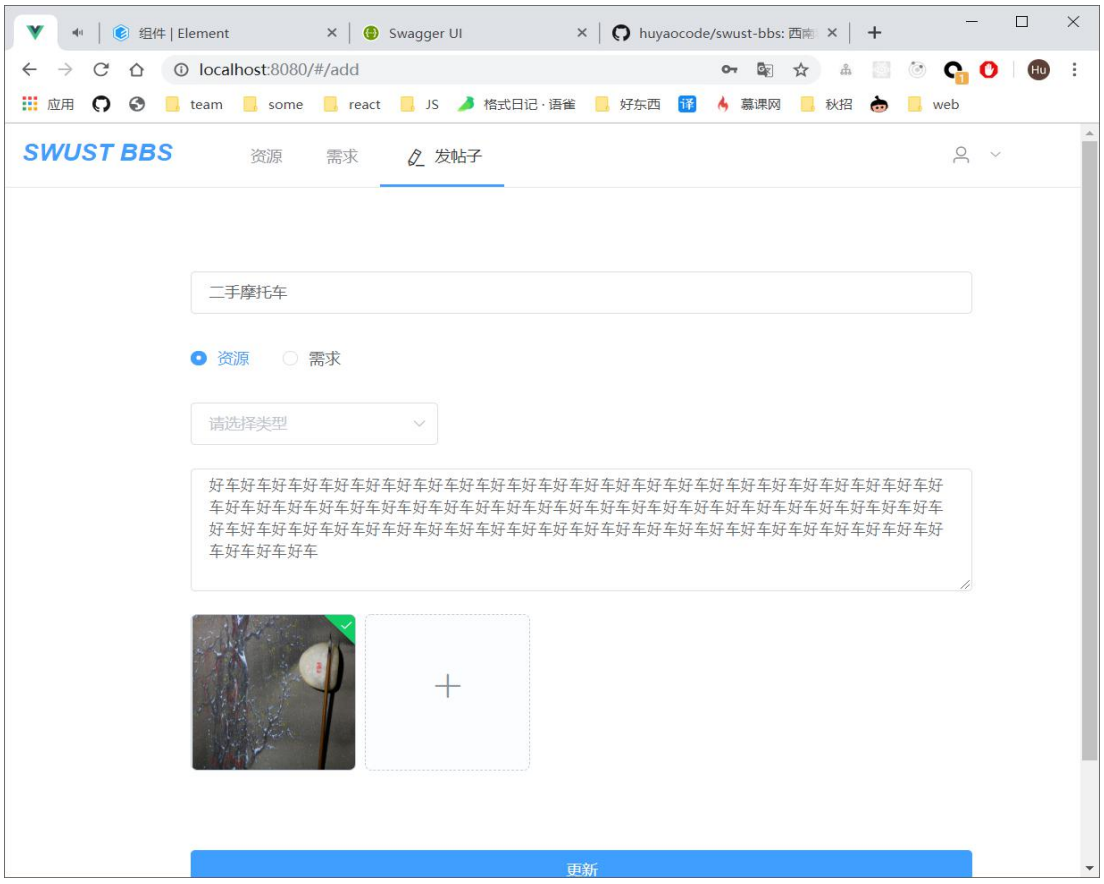
可以查看每个信息的详情界面，包括信息名称，卖家信息，信息描述等。

6. 用户信息详细表：



可以浏览用户的详细信息，包括头像、昵称、手机号、性别、学院、班级。

7. 上传信息界面：

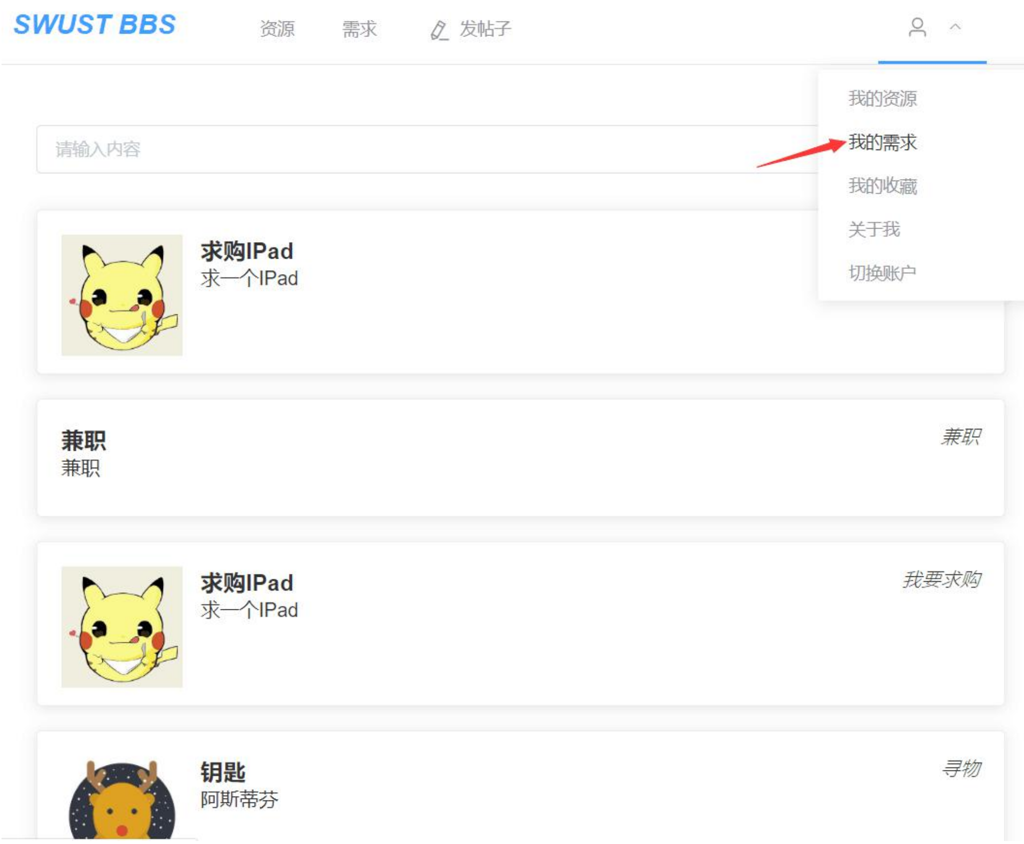


点击发帖子可以进入上传信息信息的界面，输入名字后，选择资源或者需求的标签，再选择信息的类别，最后写入简介和上传照片后就可以上传。

8. 我发布的资源信息：


SWUST BBS
资源
需求
发帖子
我的资源
我的需求
我的收藏
关于我
切换账户
请输入内容
谁的Iphone手机掉了?
在龙山
手机掉了
手机在某某地方掉了
失物招领
二手摩托车
好车
闲置转让
图片测试
他扑通扑通
技术技能

9. 我发布的需求信息：


SWUST BBS
资源
需求
发帖子
我的资源
我的需求
我的收藏
关于我
切换账户
请输入内容
求购IPad
求一个IPad
兼职
兼职
兼职
求购IPad
求一个IPad
我要求购
钥匙
阿斯蒂芬
寻物

可以查看自己发布的需求信息

10. 收藏视图：

展现用户收藏的缩略图，包括信息的描述，信息的图片，信息的标题等。

六、Spring Data JPA 复杂查询

1. InformationDao 中的查询：

```
<select id="listAll" resultMap="BaseResultMap" parameterType="com.team2.vo.cmt.InformationQuery">
select a.id,a.user_id,a.title,a.content,a.picture,a.status,a.type,a.category_id,
        a.create_time,a.update_time,a.end_time,b.name
from cmt_information a join cmt_category b on a.category_id =b.id
<trim prefix="where" prefixOverrides="and|or">
<if test="title != null and title != ''">
and (a.title like CONCAT('%',#{title},'%') or a.content like CONCAT('%',#{title},'%'))
</if>
<if test="status != null and status !=''">
       and a.status like CONCAT('%',#{status} ,'%')
</if>
```

```
<if  test="type  !=  null  and  type  !=''">
        and  a.type  like  CONCAT('%',#{type}  ,'%')
</if>
<if  test="categoryId  !=  null">
        and  a.category_id  =#{categoryId}
</if>
<if  test="name  !=  null">
        and  b.name  like  CONCAT('%',#{name},'%')
</if>
</trim>
    order  by  create_time  desc
  </select>
```

通过 Cmt_information、cmt_category 表中筛选出 id,user_id,title,content,apicture,status,category_id

2. CollectionDao 中的查询：

```
@Select("select i.* from cmt_information i ,cmt_collection c "
           + "where c.user_id=#{userId} and c.information_id=i.id")
   List<Information> getCollection(@Param("userId")Long userId);
```

通过 cmt_information,cmt_collecton 两张表中筛选出用户的信息

七、总结：

我们的项目总共实现了用户的登陆，用户的注册，查用资源列表、需求的列表，可以对资源等信息进行收藏、也可以查看信息的详情以及查看用户的详细信息。在使用 spring boot 的过程中，spring boot 可以非常方便、快速搭建项目，使我们不用关心框架之间的兼容性，适用版本等各种问题，我们想使用任何东西，仅仅添加一个配置就可以，所以使用 sping boot 非常适合构建微服务。