中科院实习周报
第一周

姓名：宋欣洋

一、实习内容

1. 学习 spring boot 框架

微服务：开发阶段的独立，不像是链式结构，有一点竖式开发的特点，面向功能而不是子系统。

架构设计：隔板模式，减少伤害。

具体实现：本周使用的 SpringBoot+mybatis+thymeleaf 来实现登录注册以及增删改查的需求。

2. 通过简单实现注册登录了解操作逻辑

用户故事：

1：用户输入用户名密码点击登录，界面跳转进入登陆后页面。

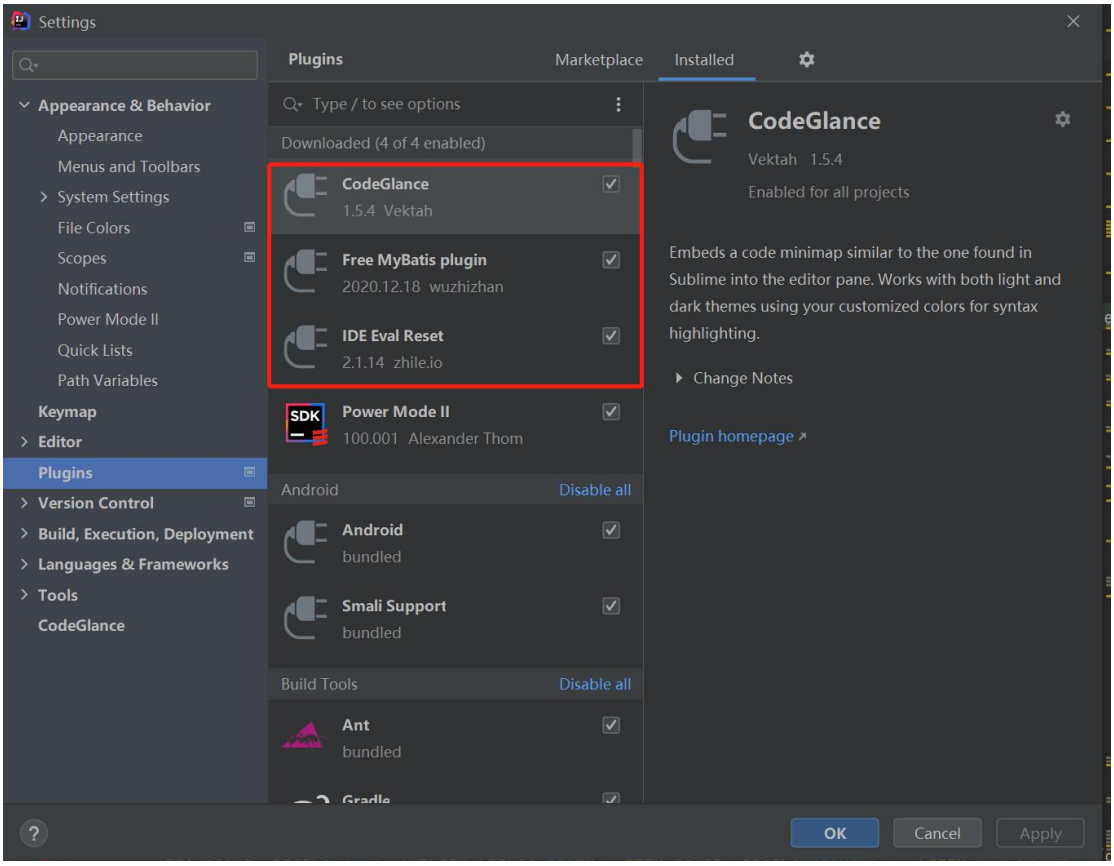
2：用户点击注册，界面跳转到注册界面，用户输入信息并提交，界面显示注册成功，返回登录页面。

插件：

CodeGlance：用于在众多的代码中便于定位，参考 vs code。

Free MyBatis：用于在 mapper.xml 中找到函数的依赖，直接定位。

IDE MyBatis plugin：用于间接破解 idea。



技术实现：

首先，登录用户故事的实现使用 form 以及 model 来传输数据到 controller。

登录后的页面可以用 model 中的 user 来识别登录的用户。
引入：<html lang="en" xmlns:th="http://www.thymeleaf.org">

```
@RequestMapping(value = "/loginIn",method = RequestMethod.POST)
public String login(Model model,String name,String password){
    UserBean userBean = userService.loginIn(name,password);
    if(userBean!=null){
        model.addAttribute( s: "user", userBean);
        return "success";
    }else {
        return "error";
    }
}
```

其次，注册的用户故事的实现判断用户是否存在
可以通过邮箱的唯一性来判断

```
@RequestMapping("toRegister")
public String toRegister(Model model){
    UserBean user = new UserBean();
    model.addAttribute( s: "user", user);
    return "register";
}

@RequestMapping("/Register")
public String Register(UserBean user){
    UserBean userEmail = userService.findUserByEmail(user.getEmail());
    if (userEmail != null){
        System.out.println("already 存在");
        return "userHave";
    }
    else {
        userService.Register(user);
        return "login";
    }
}
```

在前端的 action 执行之前，为了防止用户的蹩脚操作，尽全力阻止用户的各种操作带来的数据库的错误插入：

使用 for 语句，遍历整个表单，用 onsubmit = "return checkInput(this)"来判断是否需要提交，具体实现如下：

```
function checkInput(form) {
    var reg = new RegExp(/^\d+$/);//判断手机号是不是数字
    for (let i = 0; i < form.length; i++) {
       // alert("进来了");
        if (form.elements[i].value == "") {
            alert("请您输入" + form.elements[i].placeholder);
            //这里返回一个Boolean值，从而确定表单是否能够提交
            return false;
        }
        if(form.elements[5].value!= form.elements[6].value){
            alert("密码不一致,请修改密码！");
            return false;
        }
        if(form.elements[3].value!='2345') {
            alert("验证码错误");
            return false;
        }
        if (form.elements[1].value.length<11||form.elements[1].value.length>11||!reg.test(form.elements[1].value)){
            alert("您的手机号输入有误，请重新输入！");
            return false;
        }
    }
alert("注册中！");
return true;
};
```

数据库：

由于可以在前端判断是否用户填写的是空的，所以数据库不特意标志 not null 的元素。

```
create table user
(
    id       int auto_increment
        primary key,
    name     varchar(20) null,
    password varchar(20) null,
    tel      varchar(11) null,
    email    varchar(30) null
)
```

3. 前后端实现增删改查

首先，为管理员设置一个 table 来存储管理员的名字以及密码

管理员在登录后才可随意对用户进行增删查改

```java
@RequestMapping("toManagement")
public String toManagement() { return "management"; }


@RequestMapping("/loginInManager")
public String loginManager(Model model,String name, String password) {
    UserBean userBean = userService.loginInManager(name, password);
    List<UserBean> users = userService.findByAll();
    model.addAttribute( s: "users", users);
    if (userBean !=null){
        return "userList";
    }
    else {
        return "error";
    }
}
```

在显示用户列表界面，重点是表单的使用，合理利用 thymeleaf 来获取特定改动以及删除对象的 id，有了 id 即可利用 id 来 select 特定的 user。

```html
<tbody>
<tr th:each="user : ${users}">
<!--    each遍历users，这样得到的user是特定的user，因此可以在特定的user上做edit操作
    可以delete操作-->
  <th scope="row" th:text="${user.id}"></th>
  <td th:text="${user.name}"></td>
  <td th:text="${user.password}"></td>
  <td><a th:href="@{/toEdit(id=${user.id})}" class="ui orange basic left pointing label"><i class="clone icon"></i>edit</a></td>
  <td><a th:href="@{/delete(id=${user.id})}" class="ui orange basic left pointing label"><i class="tag icon"></i>delete</a></td>
</tr>
</tbody>
```

管理员点击修改，跳到的修改的页面，可以用 model 来获取特定用户的信息，用 th:value 直接显示到 input 的框里

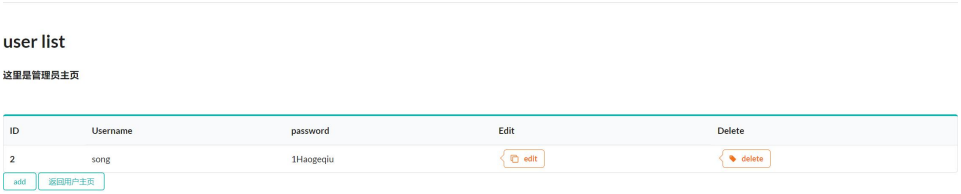
```html
<body class="container" style="text-align:center;margin-top: 100px;">
    <br/>
    <h1>修改用户</h1>
    <br/><br/>
    <div class="with:80%">
        <form  class="form-horizontal" th:action="@{/edit}" th:object="${user}" method="post">
            <input type="hidden" name="id" th:value="*{id}" />
            <div class="ui icon input" style="margin-top: 20px;">
                <input type="text" class="form-control" name="name" id="name" th:value="*{name}" placeholder="name"/>
            </div>
            <div class="ui icon input" style="margin-top: 20px;">
                <input type="text" class="form-control" name="password" id="password" th:value="*{password}" placehol
            </div>
            <input type="submit" value="确定" class="ui middle middle floated teal basic button" style="margin-top: 20
        </form>
    </div>
</body>
</html>
```

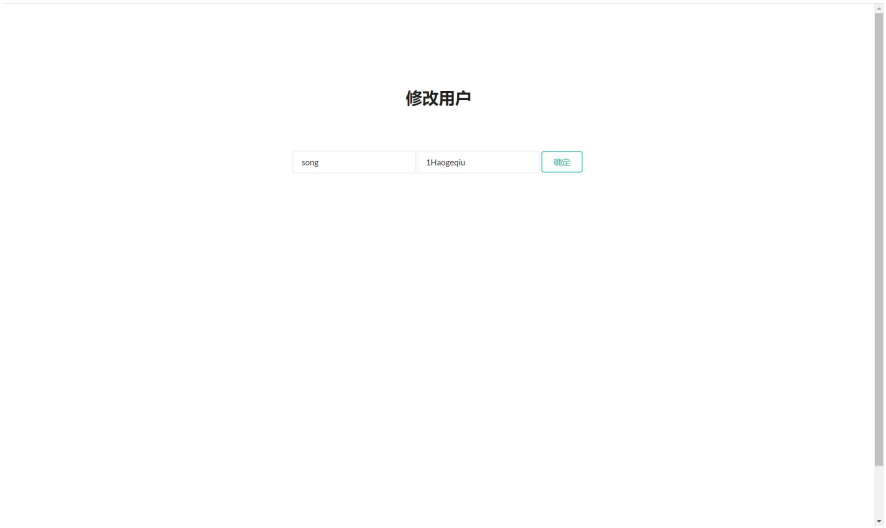
附图：

这里使用了 semantic-ui 来美化前端的 ui，类似 element-ui 直接引入直接

使用：


localhost:8080/toManagement
这里是管理员主页，请输入管理员的密码
姓名
密码
login


user list
这里是管理员主页
ID
Username
password
Edit
Delete
2
song
1Haogeqiu
edit
delete
add
返回用户主页


修改用户
song
1Haogeqiu
确定

4. 版本控制 SVN 以及 git 的学习

SVN 学习参考的网址如下：

svn 分支介绍和使用_lyf_ldh 的博客-CSDN 博客_svn 分支

为了项目小组开发的冲突减少，可以使用新建分支的方法，并将 trunk 来合并的新建的分支上，等到些许更改后，把新分支合并到 trunk 上。

注意：一定要先 add 后 commit。

Git 与 SVN 的不同在于，svn 属于可视化的操作，更加直白。

Git 与 SVN 一样有自己的集中式版本库或服务器。但，GIT 更倾向于被使用于分布式模式，也就是每个开发人员从中心版本库/服务器上 chect out 代码后会在自己的机器上克隆一个自己的版本库。

5. 升级注册以及登录+忘记密码更改密码功能

1：邮箱验证

开启邮箱 smtp 协议权限，方可作为发送方发送验证码

用 ajax 来把 data 传输到控制层：

这里 aiax 来发送 data 的时候，控制层接受 data 的时候，执行页面跳转失败，是由于不是 mvc 的架构才会如此。

```
$.ajax({
    type: "POST", //用POST方式传输
    dataType: "json", //数据格式:JSON
    url: '/sendMessage',
    data: 'email='+ document.getElementById("email").value+
        '&name='+ document.getElementById("name").value+
        '&tel='+document.getElementById("tel").value+'',
    error: function (XMLHttpRequest, textStatus, errorThrown) { },
    success: function (data){
        console.log(data[0]);
        console.log(data[1]);
    }
});
```

由控制层来实现发邮件的功能

```java
@RequestMapping(value = "/sendMessage",method=RequestMethod.POST)
public String email(Model model,String email,String name,String tel)throws Exception {
    System.out.println(email);
    UserBean user = new UserBean();
    user.setName(name);
    user.setTel(tel);
    user.setEmail(email);
    model.addAttribute( s: "user", user);
    UserBean userEmail = userService.findUserByEmail(email);
    if (userEmail != null){
        System.out.println("already 存在");
        return "userHave";
    }
    if (email == null){
        return "errorEmail";
    }
    else if (userEmail == null && email != null){
        Properties properties = new Properties();
        properties.setProperty("mail.transport.protocol", "smtp");//发送邮件协议
        properties.setProperty("mail.smtp.auth", "true");//需要验证
        //properties.setProperty("mail.debug", "true");//设置debug模式 后台输出邮件发送的过程
        Session session = Session.getInstance(properties);
        session.setDebug(true);//debug模式
        //邮件信息
        Message messgae = new MimeMessage(session);
        messgae.setFrom(new InternetAddress("2574507872@qq.com"));//设置发送人
        messgae.setText("你的验证码为：" + "2345" + "。请注意，验证码有效时间为2分钟！！！");//设置邮件内容
        messgae.setSubject("邮箱验证");//设置邮件主题
        //发送邮件
        Transport tran = session.getTransport();
        //tran.connect("smtp.qq.com", 25, "邮箱账户", "邮箱授权码");//连接到新浪邮箱服务器
        tran.connect( host: "smtp.qq.com",  port: 587,  user: "2574507872@qq.com",  password: "stwadydrnjstecgc");//连接到QQ邮箱服务器
        tran.sendMessage(messgae, new Address[]{new InternetAddress(email)});//设置邮件接收人
        tran.close();
        return "register";
    }
    return "register";
```

2：验证倒计时失效有效问题

执行 onclick 函数的同时，实现获取 input 的值传输到控制层，同时把按钮设置为不可点击

```javascript
btn = thisBtn;
btn.disabled = true; //将按钮置为不可点击
btn.value = nums+'秒重新获取';
clock = setInterval(doLoop, 1000); //一秒执行一次
```

3：密码级别判断

设为三个级别，通过 onkeyup="pwStrength(this.value)"来实时判断输入的类型，通过类型的不断叠加，而判断级别，并用不同的颜色来响应：

```
function CharMode(ch){
    if (ch>=48 && ch <=57) //数字
        return 1;
    if (ch>=65 && ch <=90) //大写字母
        return 2;
    if (ch>=97 && ch <=122) //小写
        return 4;
    else
        return 8; //特殊字符
}
```

```
function pwStrength(pwd){
    var O_color="#eeeeee";
    var L_color="#d7a1a1";
    var M_color="#d2b68b";
    var H_color="#b7eca4";
    if (pwd==null||pwd==''){
        Lcolor=Mcolor=Hcolor=O_color;
    }
    else {
        var S_level=checkStrong(pwd);
        switch(S_level) {
            case 0:
                Lcolor=Mcolor=Hcolor=O_color;
            case 1:
                Lcolor=L_color;
                Mcolor=Hcolor=O_color;
                break;
            case 2:
                Lcolor=Mcolor=M_color;
                Hcolor=O_color;
                break;
            default:
                Lcolor=Mcolor=Hcolor=H_color;
        }
    }
    document.getElementById("strength_L").style.background=Lcolor;
    document.getElementById("strength_M").style.background=Mcolor;
    document.getElementById("strength_H").style.background=Hcolor;
    return;
}
```

4：及时响应非空

通过借鉴原有项目的想法，更改代码如下：

若是用户离开 input 框，没有输入任何东西，则在输入框下方给出提示：

```
$("#name").blur(function(){
    if($(this).val().length != 0){
        $(this).parent().parent().find(".error").text("");
    }else{
        $(this).parent().parent().find(".error").text("请输入用户名");
    }
})
$("#password").blur(function(){
    if($(this).val().length != 0){
        $(this).parent().parent().find(".error").text("");
    }else{
        $(this).parent().parent().find(".error").text("请输入密码");
    }
})
$("#tel").blur(function(){
    if($(this).val().length != 0){
        $(this).parent().parent().find(".error").text("");
    }else{
        $(this).parent().parent().find(".error").text("请输入电话");
    }
})
$("#email").blur(function(){
    if($(this).val().length != 0){
        $(this).parent().parent().find(".error").text("");
    }else{
        $(this).parent().parent().find(".error").text("请输入邮箱");
    }
})
$("#passwordTwo").blur(function(){
    if($(this).val().length != 0){
        $(this).parent().parent().find(".error").text("");
    }else{
        $(this).parent().parent().find(".error").text("请输入密码");
    }
})
```

5：判断用户存在与非存在

用邮箱来标识用户，并通过 select 语句来判断用户的存在，若存在则报用户存在的网页。

```java
@RequestMapping("toRegister")
public String toRegister(Model model){
    UserBean user = new UserBean();
    model.addAttribute( s: "user", user);
    return "register";
}

@RequestMapping("/Register")
public String Register(UserBean user){
    UserBean userEmail = userService.findUserByEmail(user.getEmail());
    if (userEmail != null){
        System.out.println("already 存在");
        return "userHave";
    }
    else {
        userService.Register(user);
        return "login";
    }
}
```

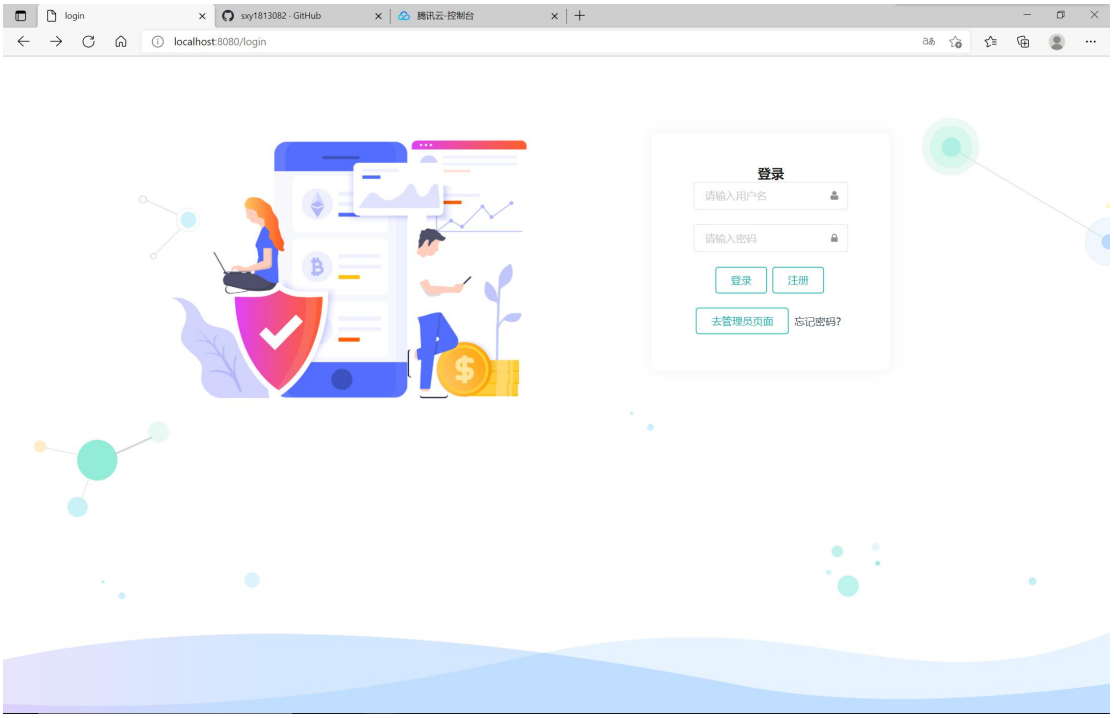
6：验证码的多样性

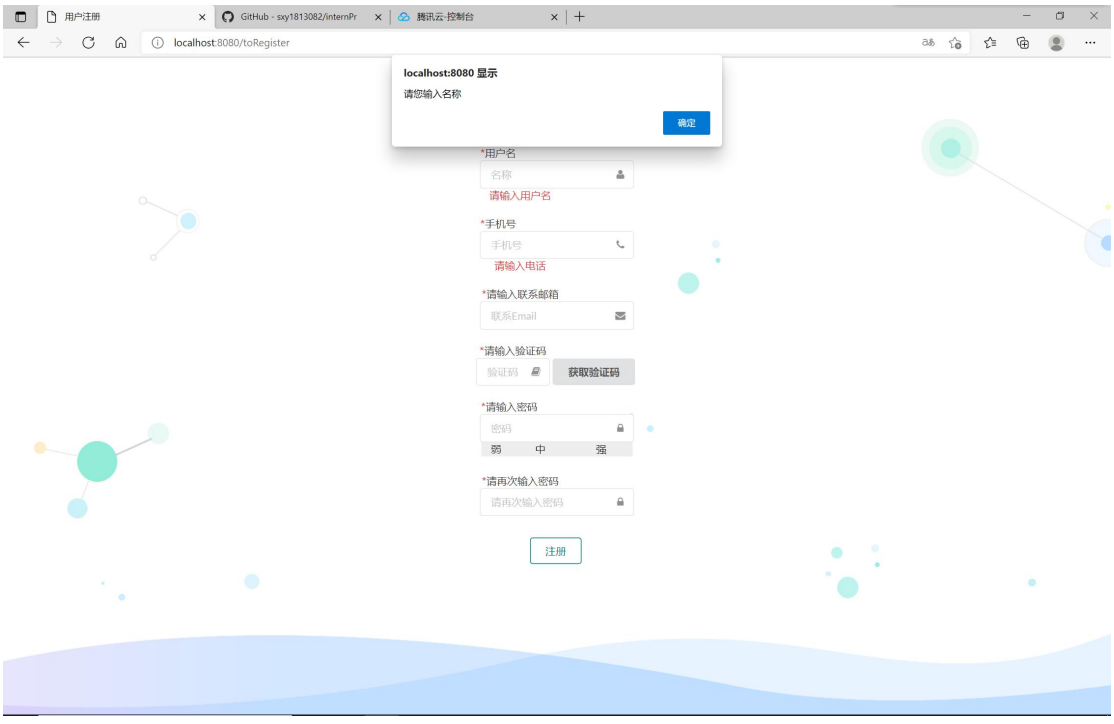
使用 visit 来实现验证码的多样性，以及验证的多样性

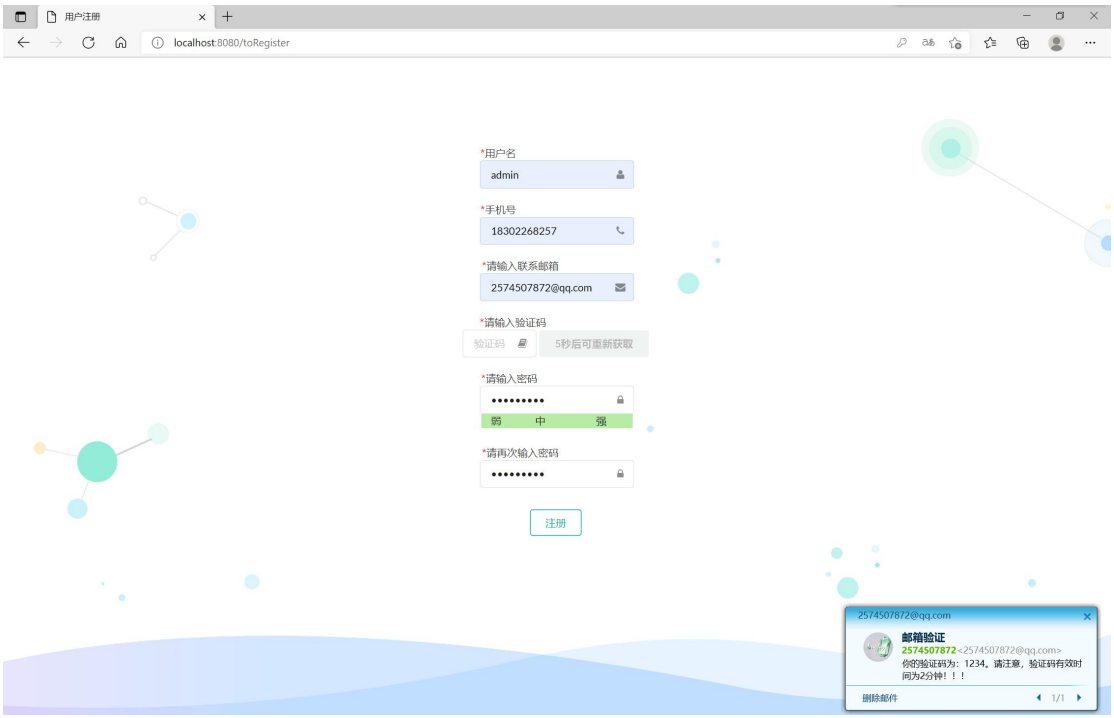
保证用户在验证码失效后会接收不一样的信息

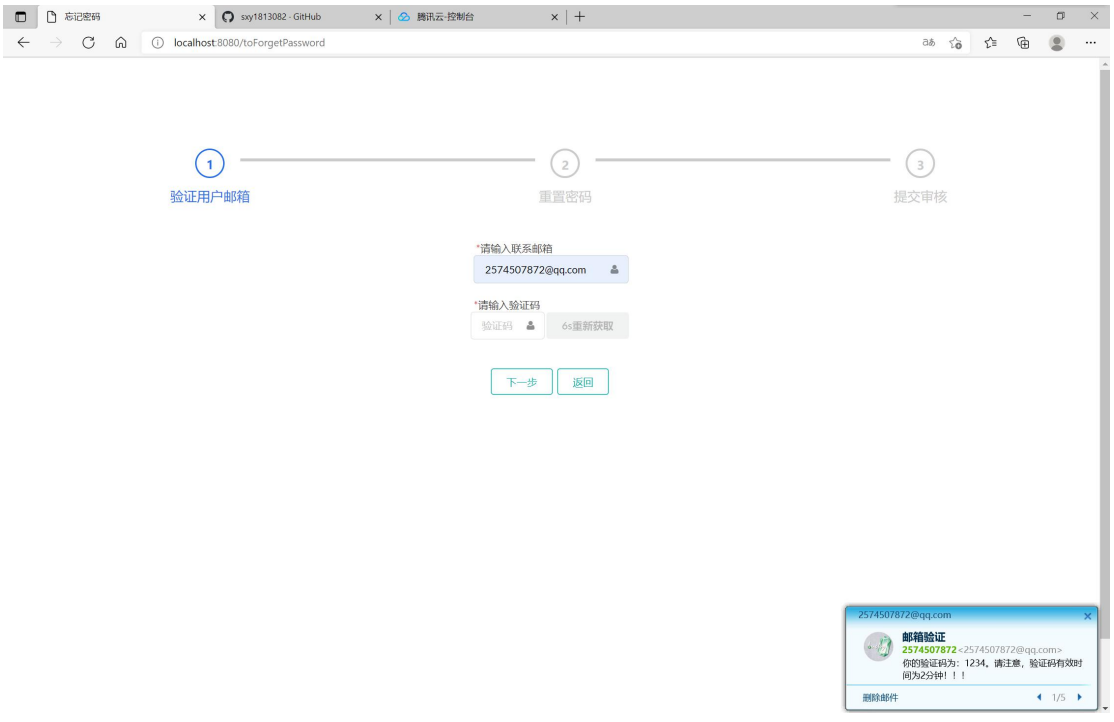
```
    $.ajax({
        type: "POST", //用POST方式传输
        dataType: "json", //数据格式:JSON
        url: '/sendMessage',
        data: 'email='+ document.getElementById("email").value+
            '&name='+ document.getElementById("name").value+
            '&tel='+document.getElementById("tel").value+
        '&yanZheng='+yanZheng[visit],
        error: function (XMLHttpRequest, textStatus, errorThrown) { },
        success: function (data){
            console.log(data[0]);
            console.log(data[1]);
        }
    });
}
if (visit == 1){
    visit = 0;
}
else {
    visit = 1;
}
```

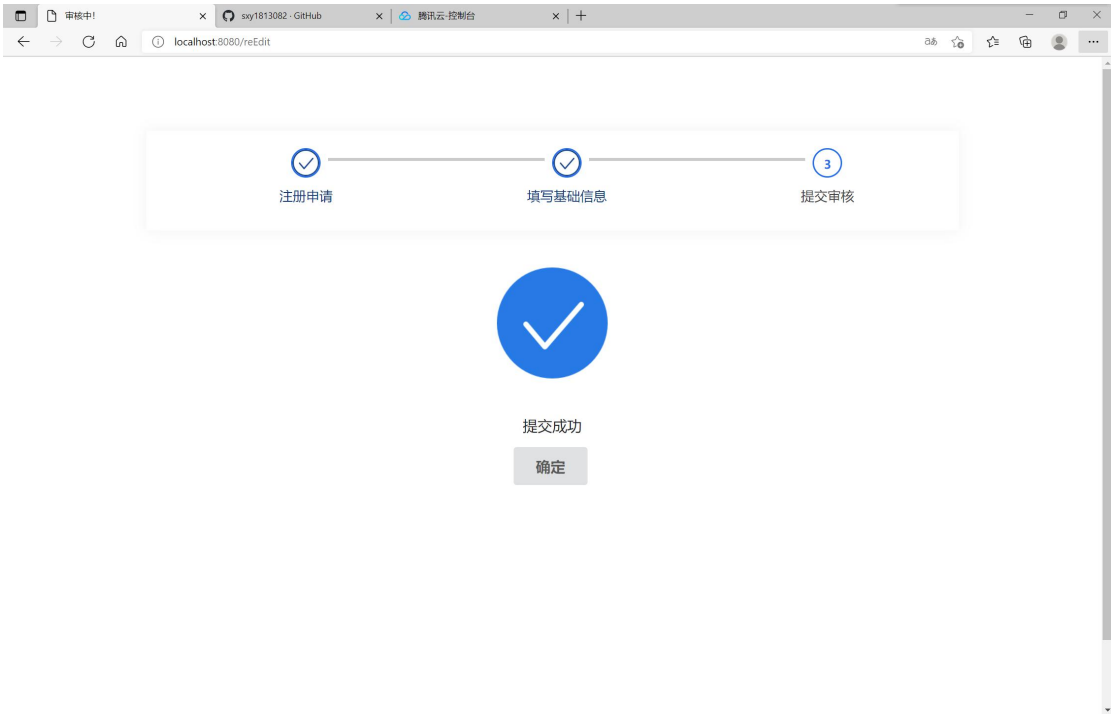
附图：


login
sxy1813082 - GitHub
腾讯云-控制台
localhost:8080/login
登录
请输入用户名
请输入密码
登录
注册
去管理员页面
忘记密码?


用户注册
GitHub - sxy1813082/internPr
腾讯云-控制台
localhost:8080/toRegister
localhost:8080 显示
请您输入名称
确定
*用户名
名称
请输入用户名
*手机号
手机号
请输入电话
*请输入联系邮箱
联系Email
*请输入验证码
验证码
获取验证码
*请输入密码
密码
弱
中
强
*请再次输入密码
请再次输入密码
注册


用户注册
localhost:8080/toRegister
*用户名
admin
*手机号
18302268257
*请输入联系邮箱
2574507872@qq.com
*请输入验证码
验证码
5秒后可重新获取
*请输入密码
弱 中 强
*请再次输入密码
注册
2574507872@qq.com
邮箱验证
2574507872<2574507872@qq.com>
你的验证码为：1234。请注意，验证码有效时间为2分钟！！！
删除邮件
1/1


忘记密码
sxy1813082 · GitHub
腾讯云·控制台
localhost:8080/toForgetPassword
1
验证用户邮箱
2
重置密码
3
提交审核
*请输入联系邮箱
2574507872@qq.com
*请输入验证码
验证码
6s重新获取
下一步
返回
2574507872@qq.com
邮箱验证
2574507872<2574507872@qq.com>
你的验证码为：1234。请注意，验证码有效时间为2分钟！！！
删除邮件
1/5

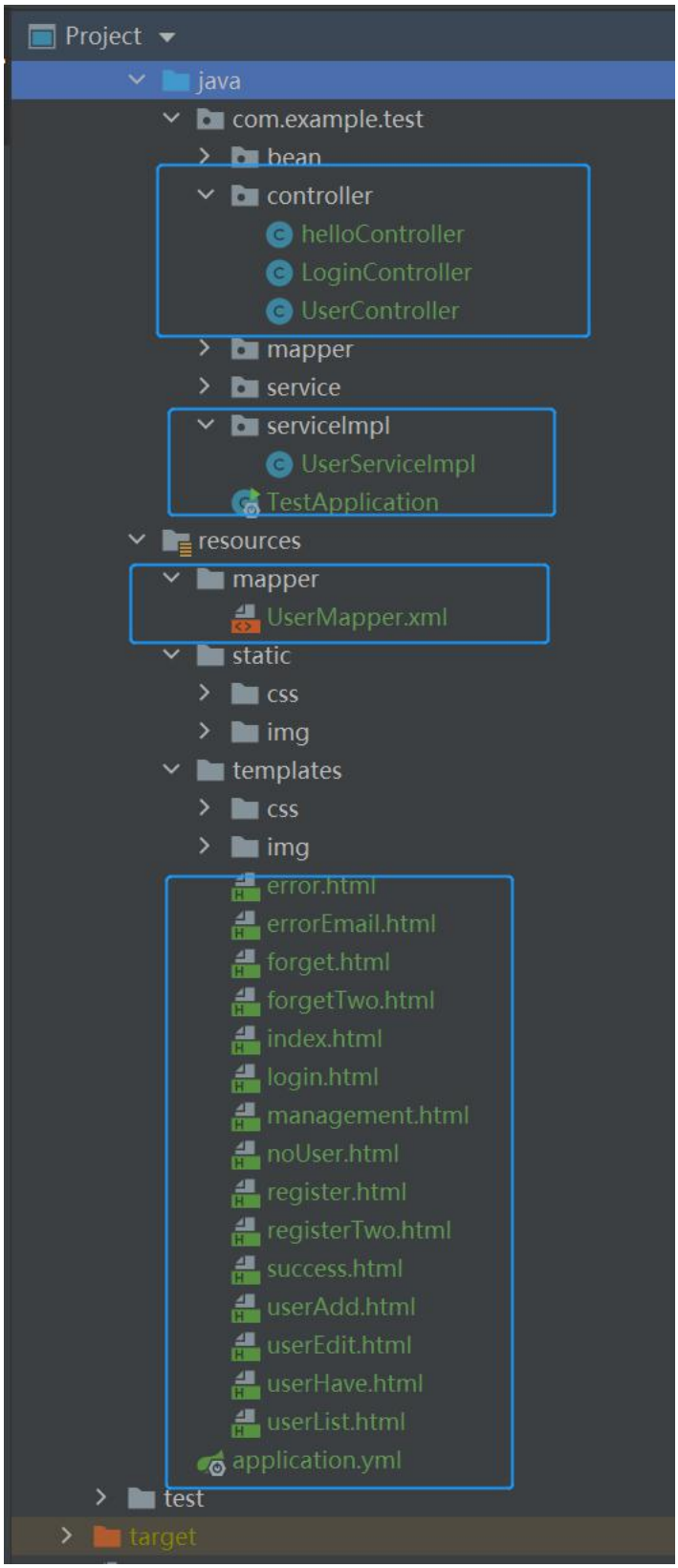


二、项目总结

1. 代码框架

   代码的主要后端在控制层以及 mapper 的 xml 文件中。

   前端主要是资源文件中的 html 文件以及静态文件夹中的 css 样式，js 主要直接写在 html 文件中：



2. 代码提交

项目上传到了 github 中以供以后总结：

https://github.com/sxy1813082/internPr.git

三、下周计划

1. 周一完善注册，优化验证码多样性，不是仅用 visit 判断。
2. 周二至周五根据导师项目选题继续学习与 springboot 有关的知识。
3. 同时每天在学习阶段中记录学习内容形成笔记，供以后查阅。