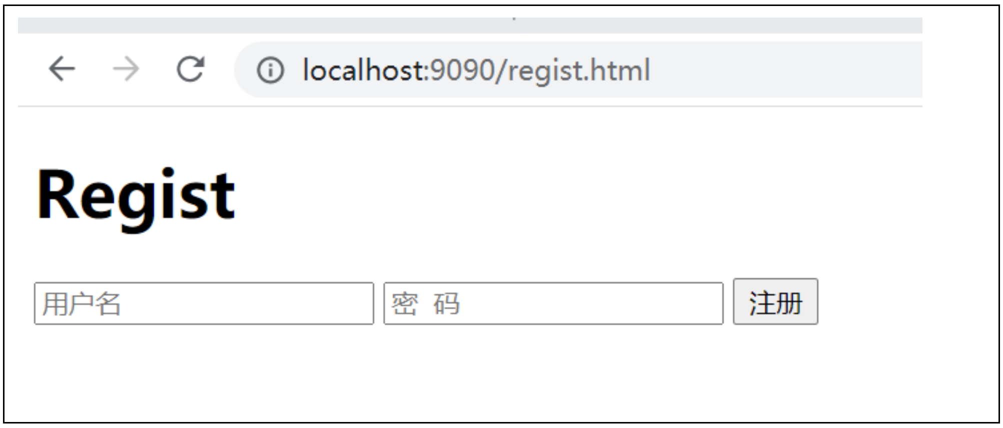
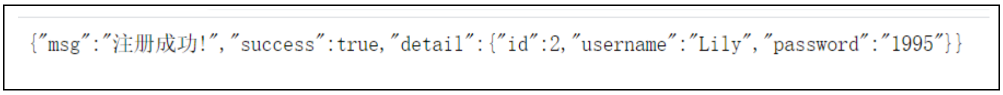
一．前端界面

(1)　注册功能

- 注册界面：

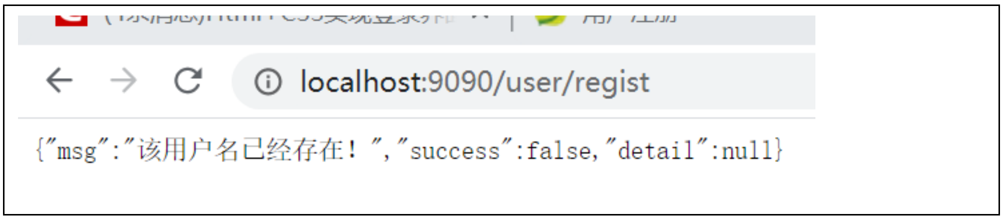


- 注册成功界面：



```
{"msg":"注册成功!","success":true,"detail":{"id":2,"username":"Lily","password":"1995"}}
```

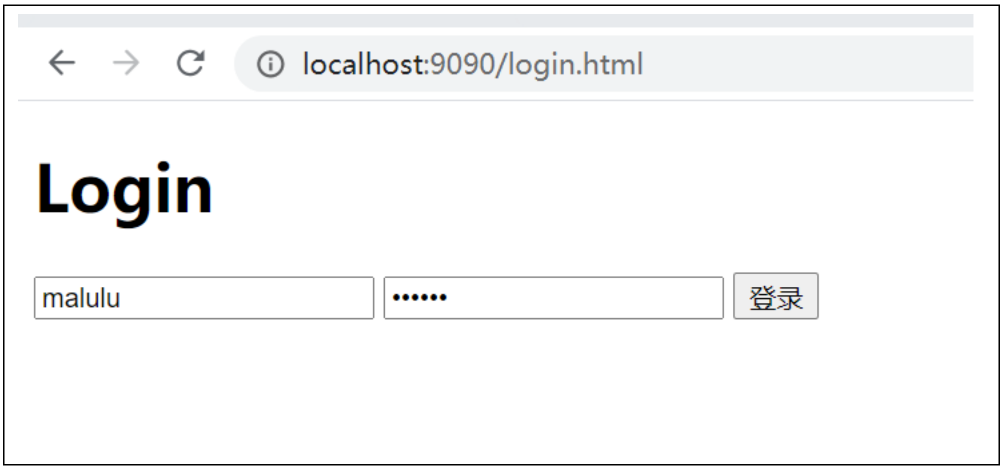
- 注册失败界面：（用户已存在）



```
{"msg":"该用户名已经存在！","success":false,"detail":null}
```

(2)　登录功能

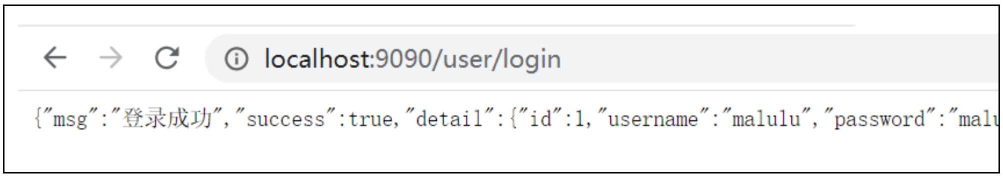
- 登录界面：



malulu

••••••

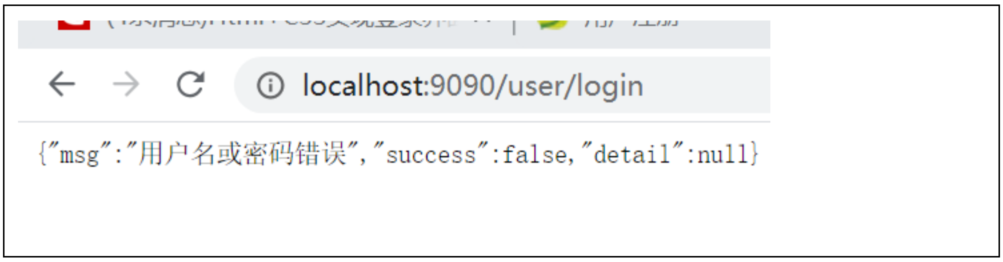
- 登录成功界面



localhost:9090/user/login

```
{"msg":"登录成功","success":true,"detail":{"id":1,"username":"malulu","password":"malu
```

- 登录失败界面



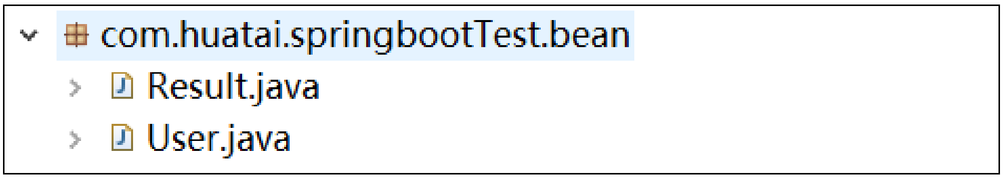
localhost:9090/user/login

```
{"msg":"用户名或密码错误","success":false,"detail":null}
```

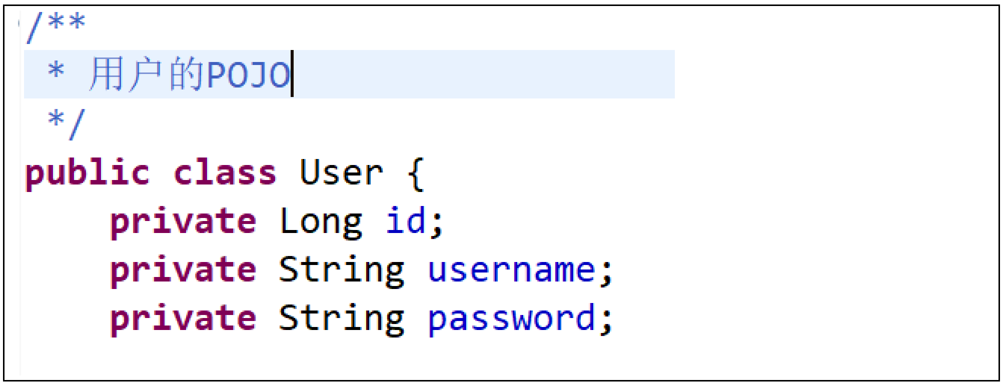
二．后端代码

（1） Model 层

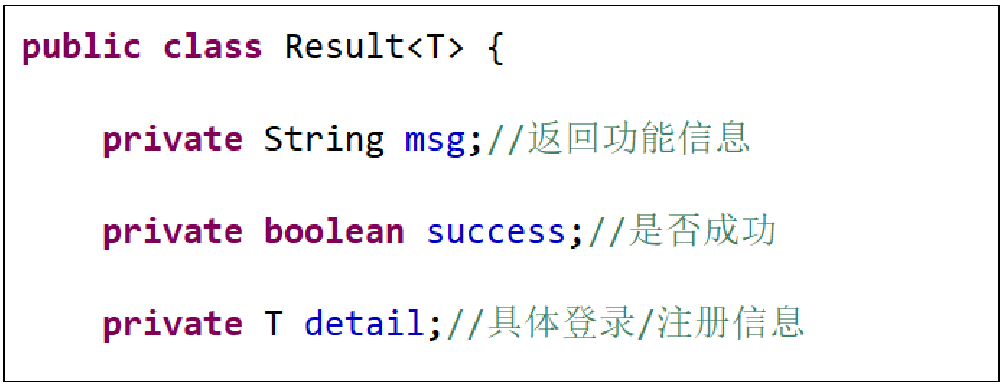
- POJO 实体类



```
com.huatai.springbootTest.bean
    Result.java
    User.java
```

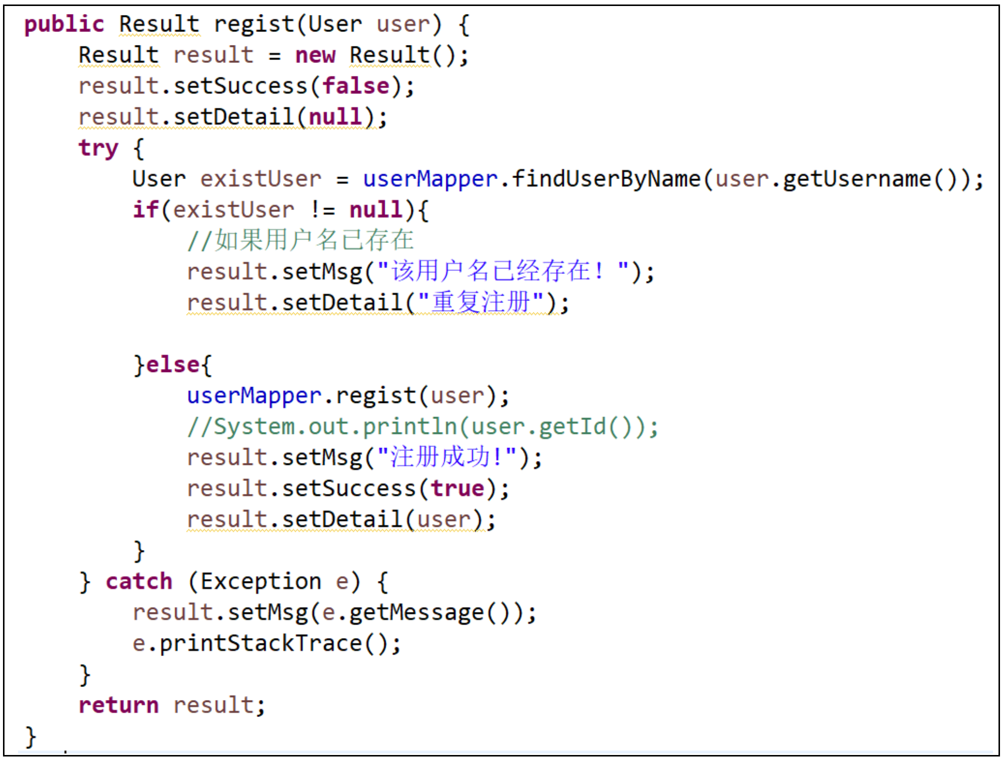


```java
/**
 * 用户的POJO
 */
public class User {
    private Long id;
    private String username;
    private String password;
```

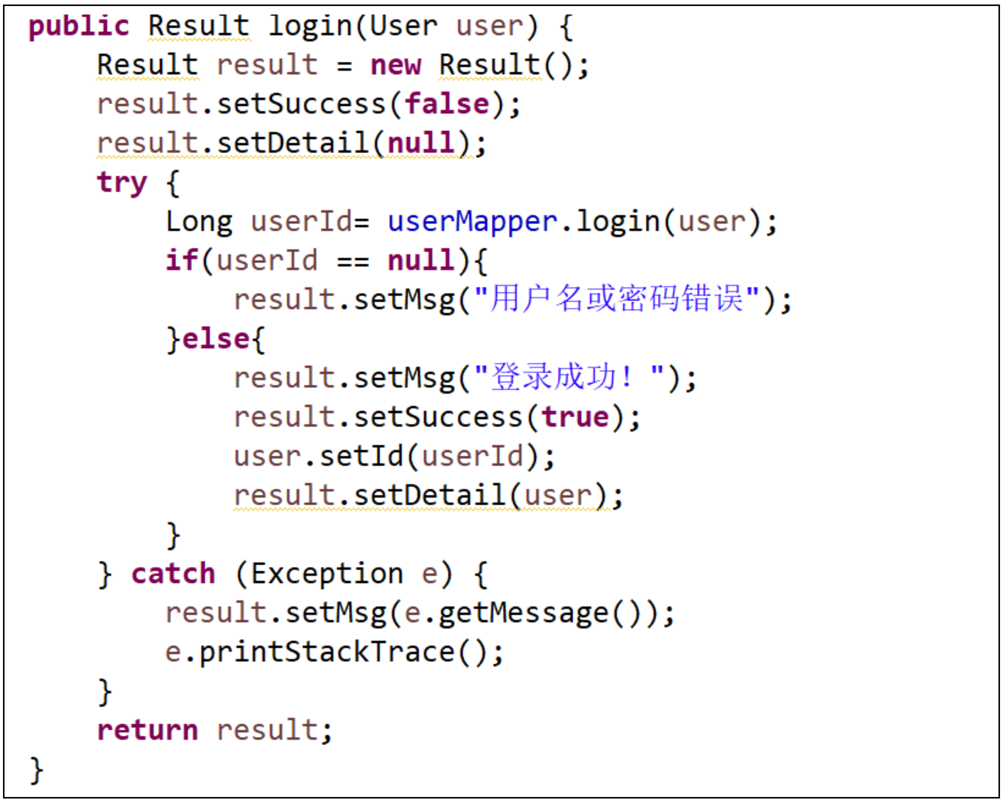


```java
public class Result<T> {

    private String msg;//返回功能信息

    private boolean success;//是否成功

    private T detail;//具体登录/注册信息
```

- Service 类

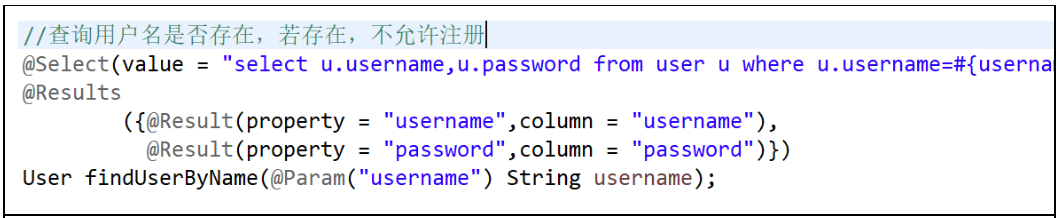


```java
public Result regist(User user) {
    Result result = new Result();
    result.setSuccess(false);
    result.setDetail(null);
    try {
        User existUser = userMapper.findUserByName(user.getUsername());
        if(existUser != null){
            //如果用户名已存在
            result.setMsg("该用户名已经存在！");
            result.setDetail("重复注册");

        }else{
            userMapper.regist(user);
            //System.out.println(user.getId());
            result.setMsg("注册成功!");
            result.setSuccess(true);
            result.setDetail(user);
        }
    } catch (Exception e) {
        result.setMsg(e.getMessage());
        e.printStackTrace();
    }
    return result;
}
```

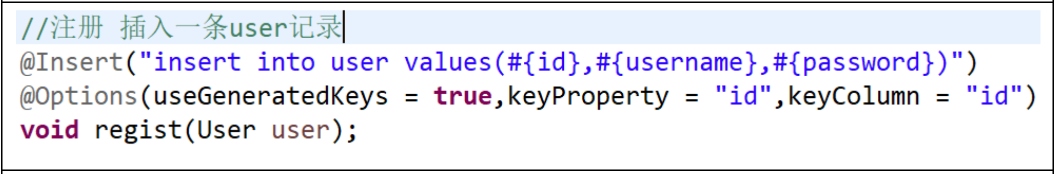


```java
public Result login(User user) {
    Result result = new Result();
    result.setSuccess(false);
    result.setDetail(null);
    try {
        Long userId= userMapper.login(user);
        if(userId == null){
            result.setMsg("用户名或密码错误");
        }else{
            result.setMsg("登录成功！");
            result.setSuccess(true);
            user.setId(userId);
            result.setDetail(user);
        }
    } catch (Exception e) {
        result.setMsg(e.getMessage());
        e.printStackTrace();
    }
    return result;
}
```

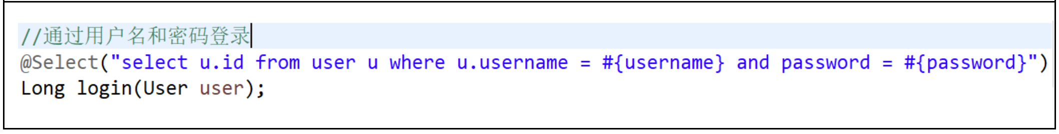
(2)　　Mapper 层



```
//查询用户名是否存在，若存在，不允许注册
@Select(value = "select u.username,u.password from user u where u.username=#{userna
@Results
        ({@Result(property = "username",column = "username"),
          @Result(property = "password",column = "password")})
User findUserByName(@Param("username") String username);
```

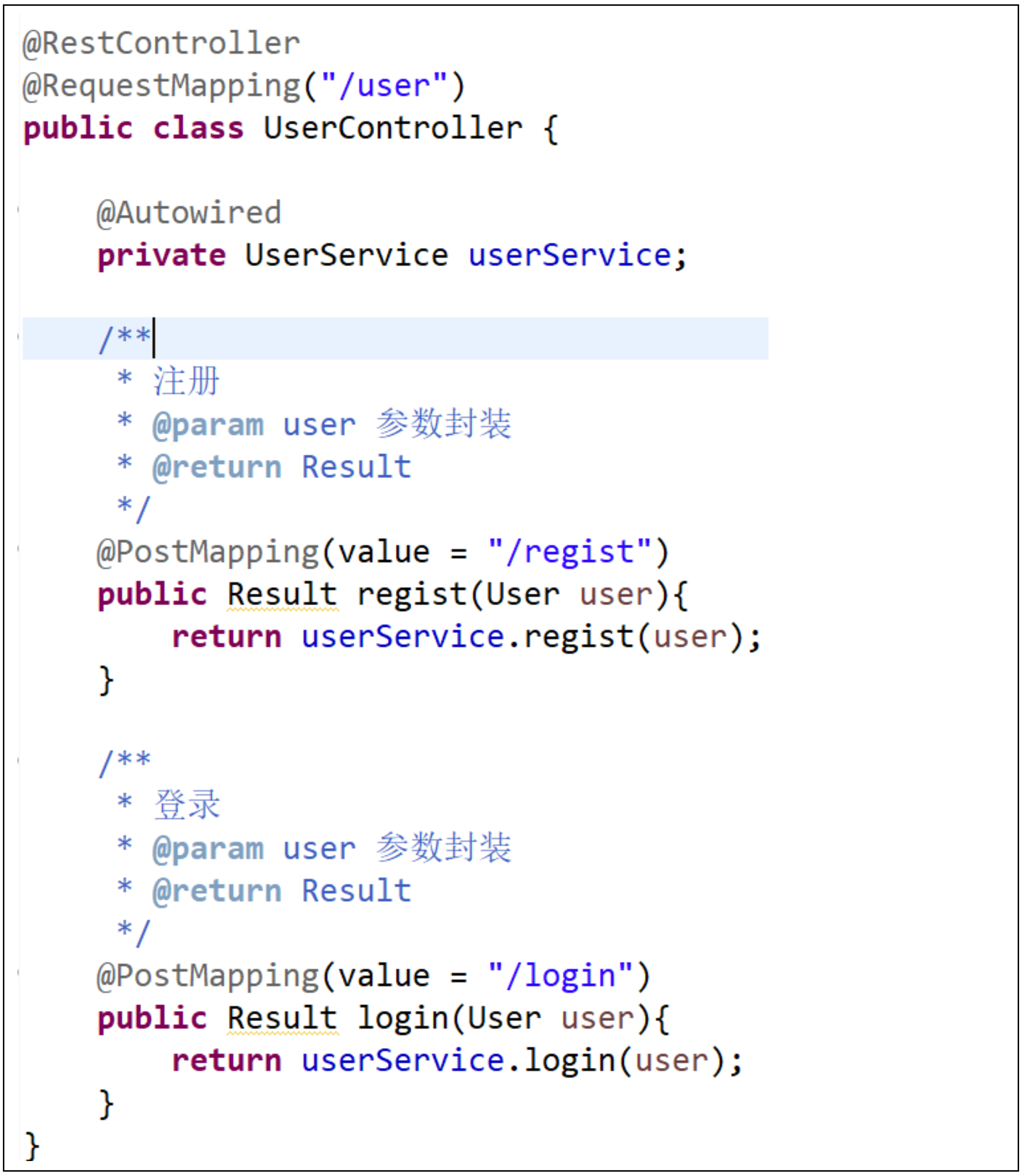


```
//注册 插入一条user记录
@Insert("insert into user values(#{id},#{username},#{password})")
@Options(useGeneratedKeys = true,keyProperty = "id",keyColumn = "id")
void regist(User user);
```



```
//通过用户名和密码登录
@Select("select u.id from user u where u.username = #{username} and password = #{password}")
Long login(User user);
```

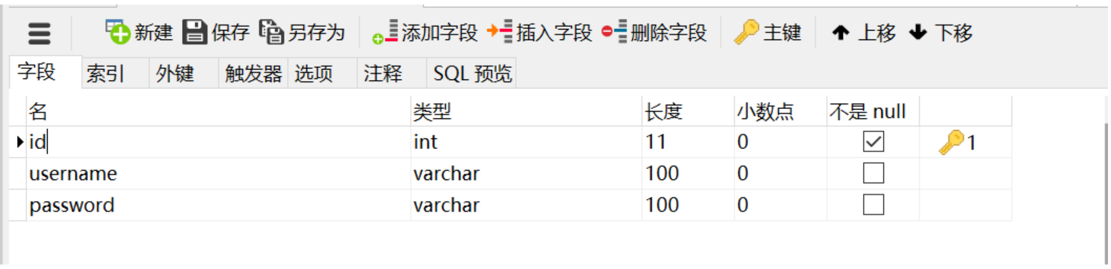
(3)　　Controller 层



```
@RestController
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserService userService;

    /**
     * 注册
     * @param user 参数封装
     * @return Result
     */
    @PostMapping(value = "/regist")
    public Result regist(User user){
        return userService.regist(user);
    }

    /**
     * 登录
     * @param user 参数封装
     * @return Result
     */
    @PostMapping(value = "/login")
    public Result login(User user){
        return userService.login(user);
    }
}
```

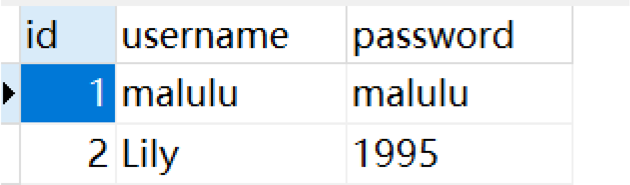
三．数据库

用户表结构:



名	类型	长度	小数点	不是 null	
id	int	11	0	☑	🔑1
username	varchar	100	0	□	
password	varchar	100	0	□	

表的所有记录:



id	username	password
1	malulu	malulu
2	Lily	1995