《综合实训三》

李林

（2020 — 2021 学年 第 1 学期）

## 第一章 模块需求描述

### 用户模块

* 前端：用户登录，用户注册，用户密码找回，用户页面展示，用户信息展示以及修改，用户余额展示
* 前端后台：用户信息增删改查
* 后端：用户登录接口，用户注册接口，用户密码找回接口，邮件发送接口，余额查询修改接口

### 商品模块

* 前端商品显示，添加购物车，购买，评论显示，访问统计，库存统计
* 后端商品增删改查，商品标签分类，商品的上架与下架

### 购物车模块

* 前端购物车显示，删除，结算，多商品结算，数量修改，总价统计，数量统计
* 后端购物车删除，信息修改，查询

### 订单模块

* 前端用户下单生成订单，购物车页面结算生成订单
* 前端订单信息展示，订单分类，订单操作
* 后端订单删除，信息修改，查询

### 数据持久层模块

* 使用数据库连接池对数据库连接进行管理，以及sql语句的执行
* 统一化对数据库的增删改查

### 视图控制模块

* 对视图进行精确控制定位
* 对请求进行分配处理

### 邮件模块

* 对用户注册，用户密码找回，用户密码修改进行邮箱引导

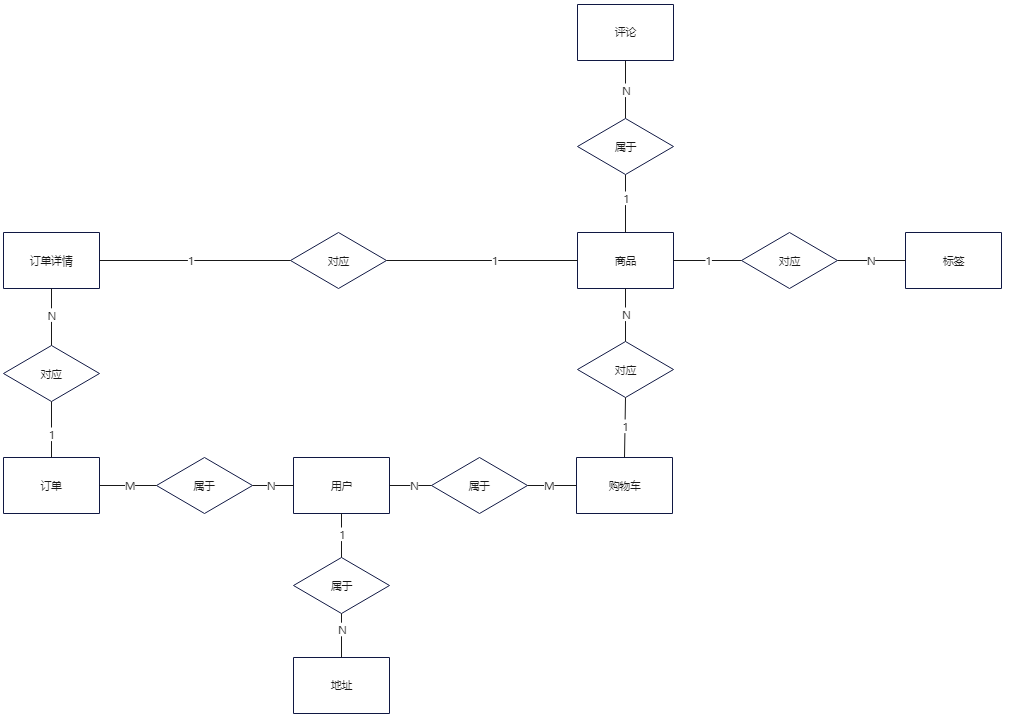
## 第二章 数据库设计

物理设计结果（表结构，并用文字简要说明作用） 表间关系

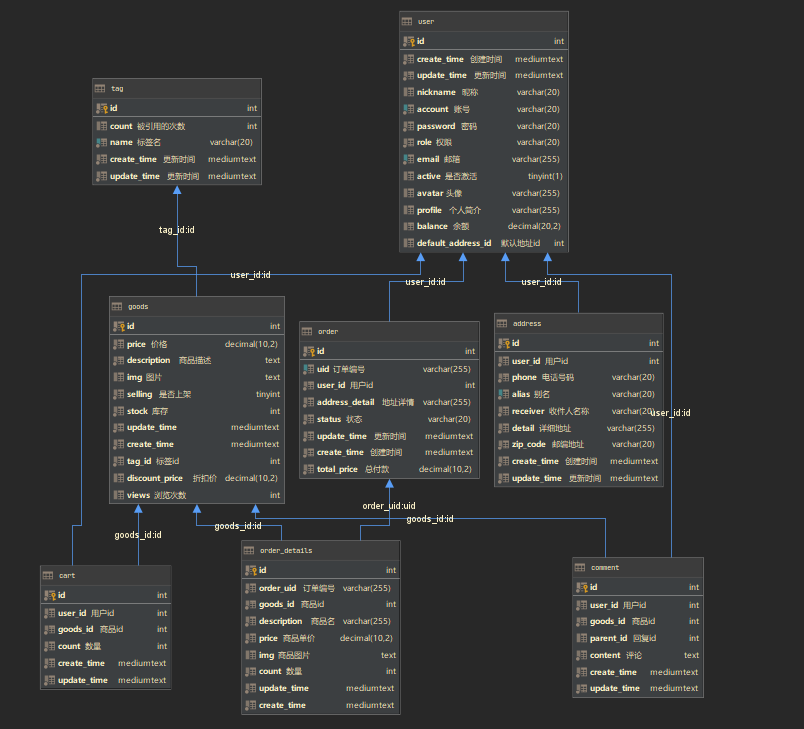
数据库主键字段为 id 且自增

数据库表中有2个公共字段 create\_time 创建时间 和 update\_time 更新时间

### ER 图



UML 依赖图



### 用户表

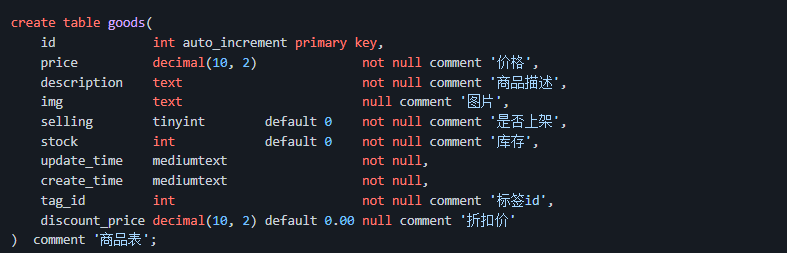


### 地址表

用于订单地址显示

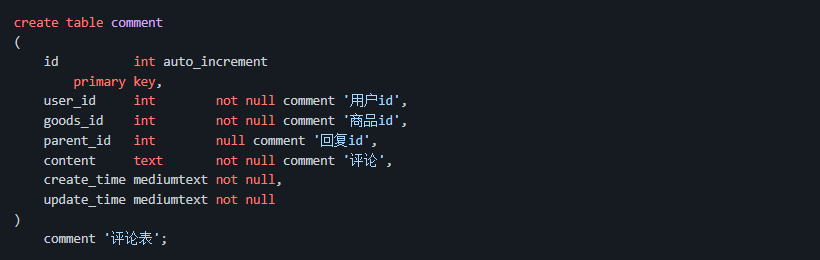


### 商品表



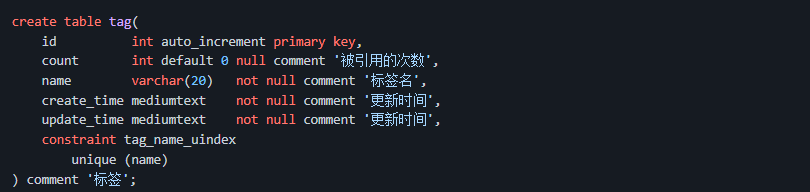
### 评论表

用于展示商品的评论

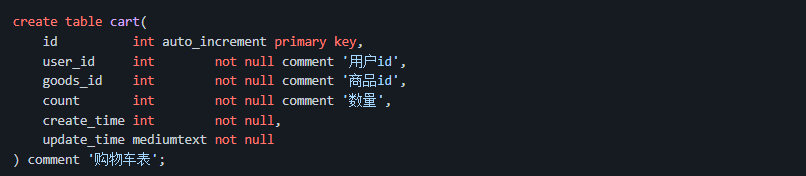


### 标签表

商品的标签分类

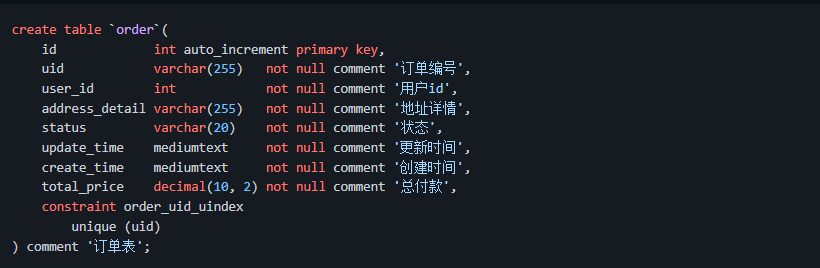


### 购物车表



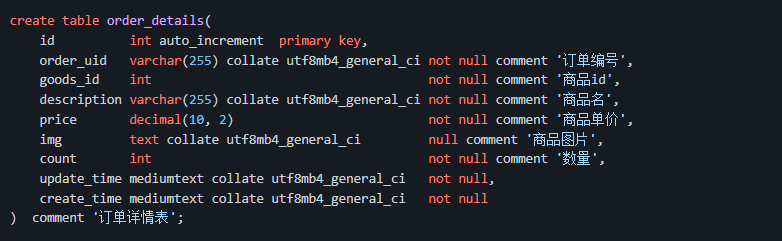
### 订单表

订单详情的主表



### 订单详情表

使用order\_uid与订单主表绑定，从而实现单个订单多个商品的功能



## 第三章 系统总体设计

（MVC结构的描述及搭建介绍）

### 系统架构

MVC： 业务模型层，视图层，控制层，也就是经典的 MVC 模式。

### 业务模型层

主要技术: mysql8.0 ，数据持久化框架 ， JavaBean , pojo

### 视图层

主要技术：jsp , js ,css

前端框架: jq , bootstrap

### 控制层

主要技术：servlet , filter , 视图控制器

### 系统搭建介绍

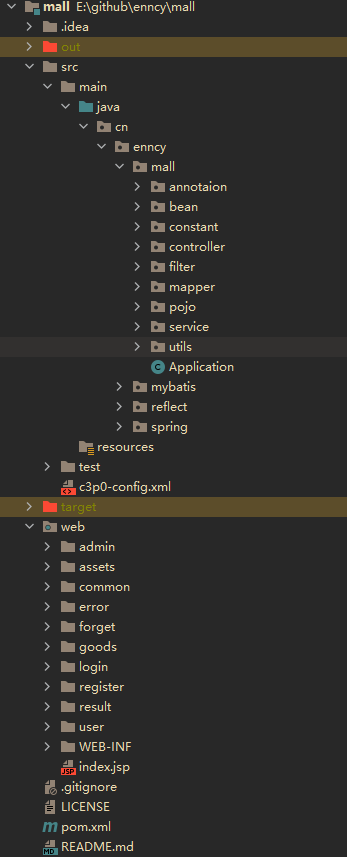
编辑器 : IntelliJ IDEA

包管理器 : maven

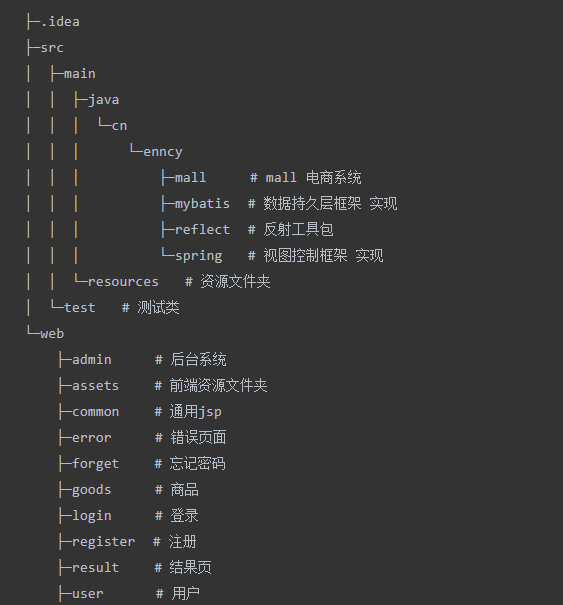
代码版本控制系统 : git

web容器 : tomcat

项目结构图



项目文件夹解析



## **第四章** 模块设计与实现

（每个功能的流程描述，相关重要代码、文件、运行截图）

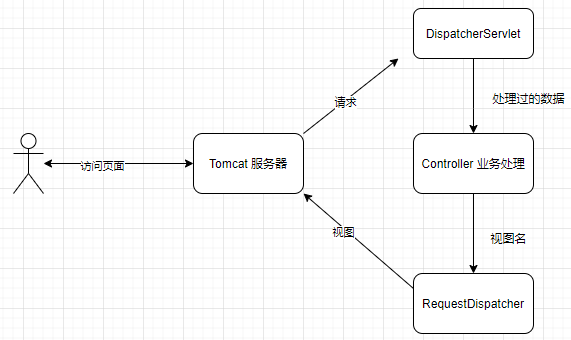
### 视图控制模块

该模块主要由 DispatcherServlet 实现 , 并继承了 Filter 接口，使之具有web过滤器的功能

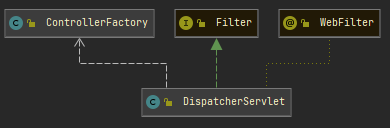
#### 流程描述

1. 反射获取带有相关注解的控制器类
2. 获取请求路径，对比控制器上的注解找出相应的控制器，以及方法
3. 解析请求体，并将参数传到，方法执行
4. 获取方法返回值，并处理，通过 getRequestDispatcher 方法分发到视图，并展示给用户

模块设计图

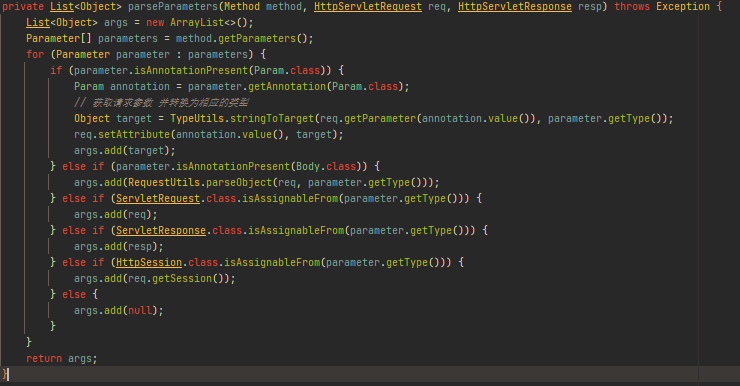


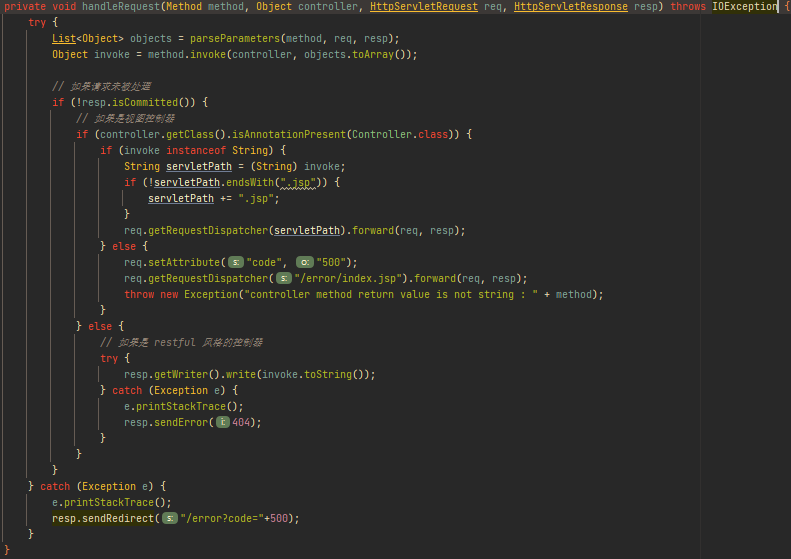
UML 类图



#### 主要代码

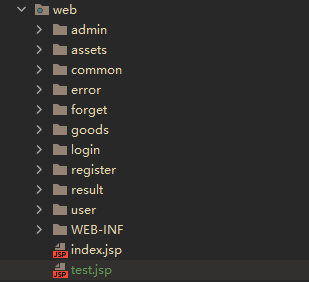
1. 在 init 初始化方法中，进行包扫描，获取项目中具有 @Controller 以及 @RestController 注解的类
2. 遍历初始化中获取到的控制器集合，并找到响应的控制器，并对控制器的属性进行修改，使得每次执行的 request 以及 response 都与当前的请求上下文致。
3. 解析请求体，并生成对应的参数，传到方法中执行

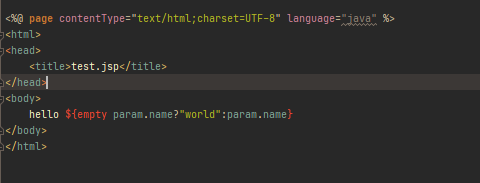
* 

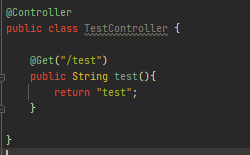
1. 处理返回值，如果是 Controller 则解析视图，如果是 RestContoller 则返回 json 数据

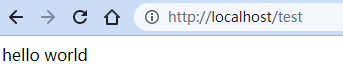
#### 运行截图

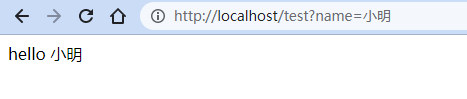
Controller 视图控制器测试



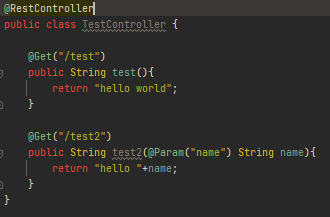


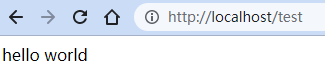


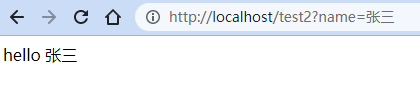


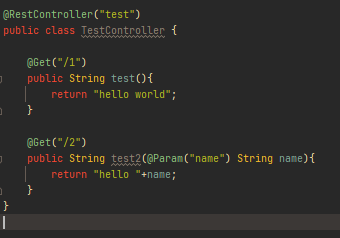


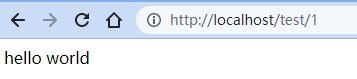
RestController rest风格的控制器测试

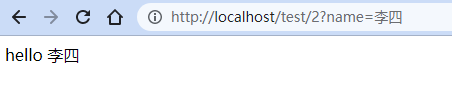












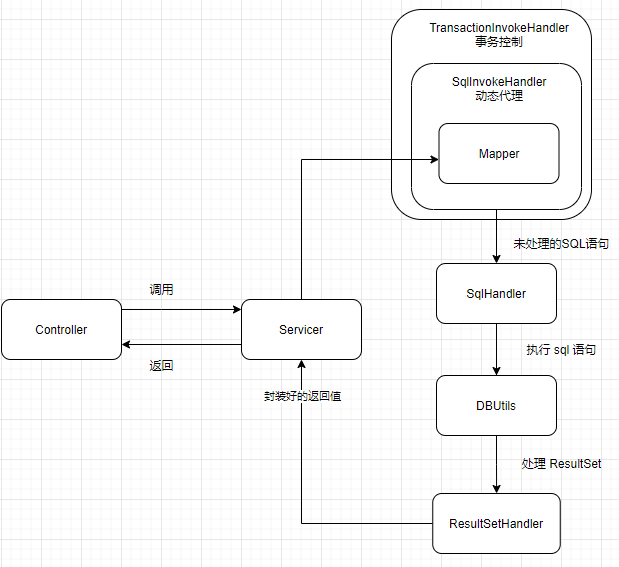
### 数据持久层模块

该模块主要由动态代理实现 , 使用 jdk 自带的动态代理接口 InvocationHandler 实现对方法的动态代理以及 sql 执行

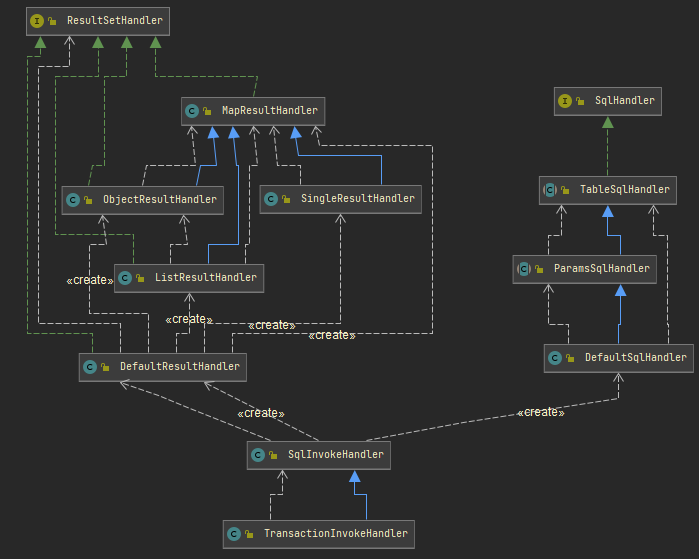
#### 流程描述

1. 创建 SqlInvokeHandler 并继承 InvocationHandler ，在业务层调用方法的时候动态代理并反射获取到对应的 sql 语句
2. 创建 TransactionInvokeHandler 对方法进行事务控制，回滚，提交
3. 对 sql 语句进行处理
4. 使用 DBUtils 进行数据库连接的获取以及管理，保证统一请求中的事务共享
5. 使用 ResultSetHandler 对查询出来的结果进行处理，并且返回给业务层

模块设计图

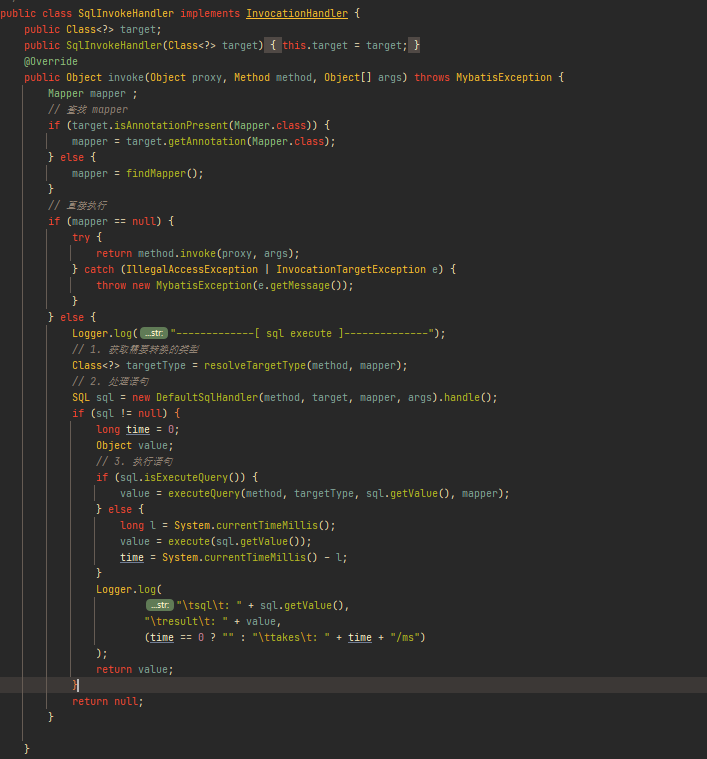


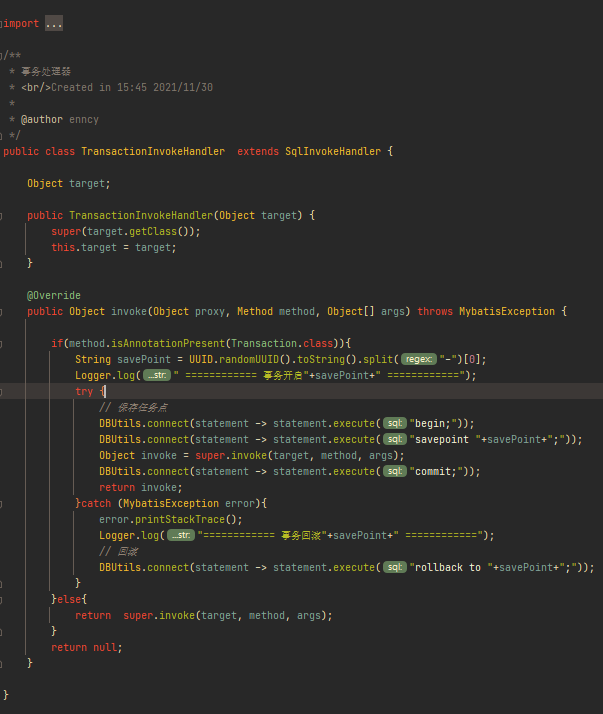
UML 类图

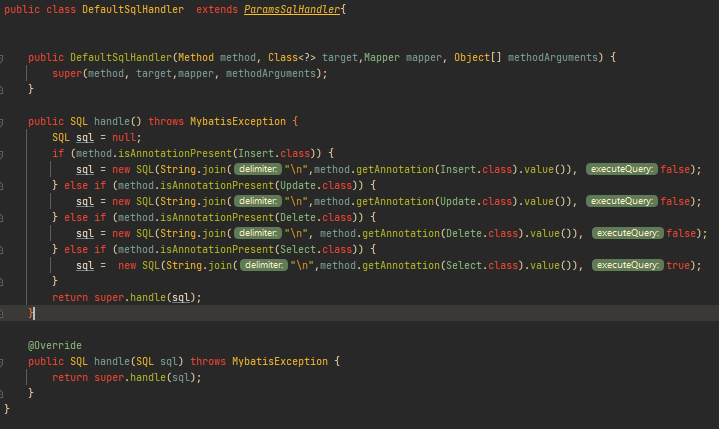
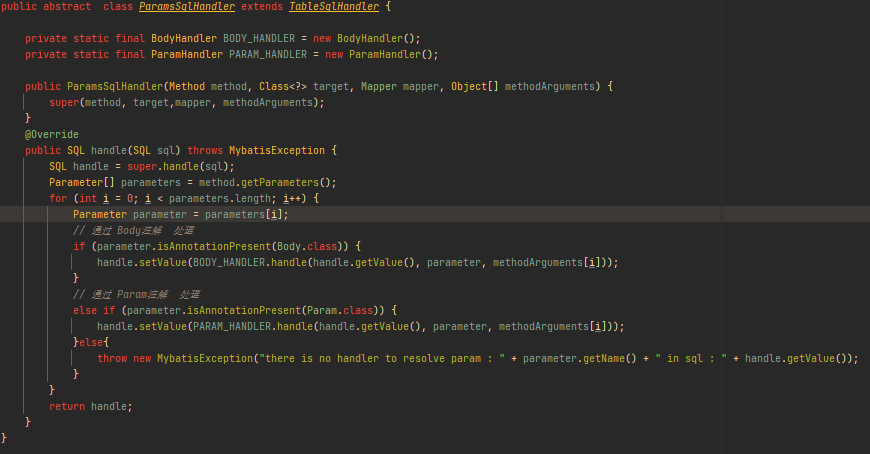


#### 主要代码

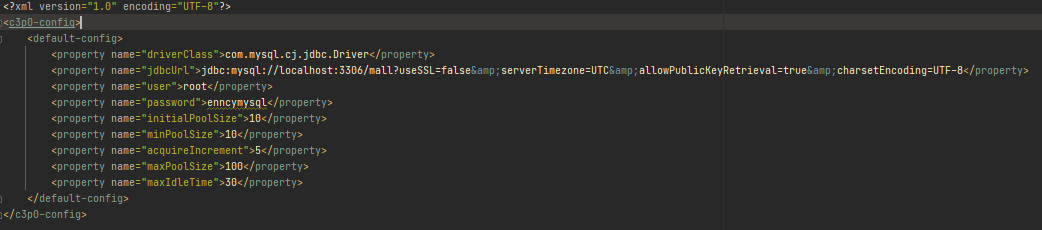
1. sql处理器，首先对代理的对象进行注解查找 ，如果存在指定的 @Mapper 注解，则此对象为持久层对象，进行sql处理执行，否则对此方法直接执行

* 

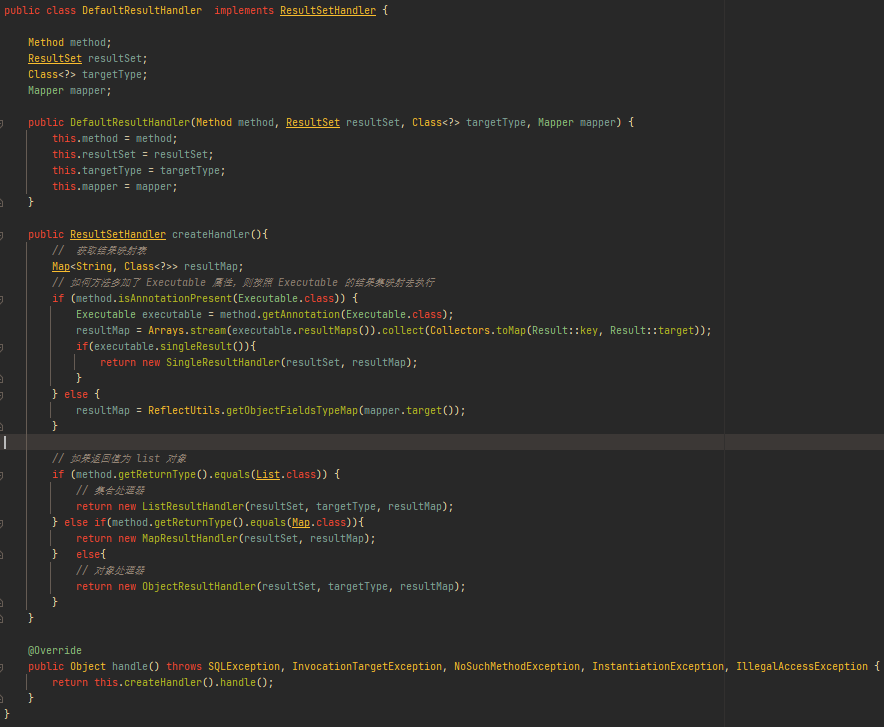
1. 事务控制处理器，使用 @Transaction 注解对指定的方法进行事务控制，并在动态代理的时候进行执行。
2. 使用字符串关键字，以及 @Param 和 @Body 注解处理sql语句，并进行替换

* 
* 

1. 使用 DBUtils 进行数据库连接，C3P0 进行数据库连接池的处理，ThreadLocal 对线程变量进行添加获取删除。

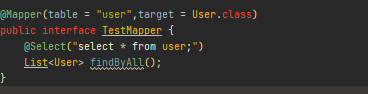
* 
* 

1. 使用 ResultSetHandler 进行返回值处理

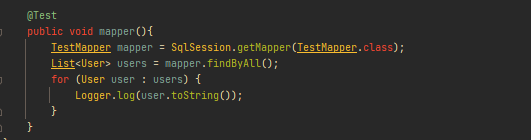
* 

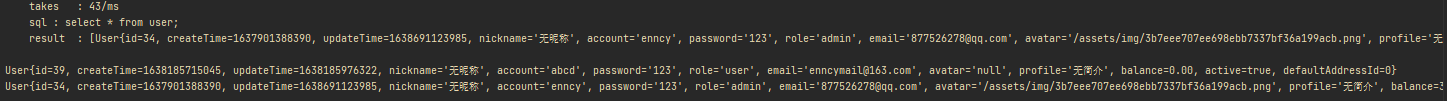
#### 运行截图

Mapper 测试类



测试方法



控制台输出

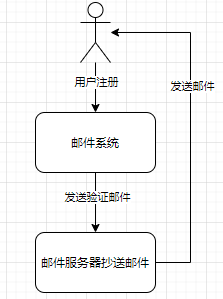
### 邮件模块

邮件模块使用java自带的 javax.mail 包实现，使用 smtp 协议进行邮件发送

#### 流程描述

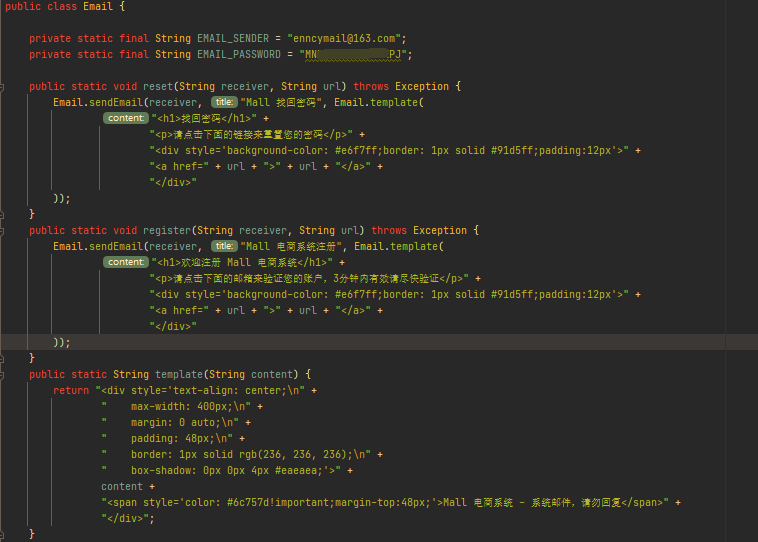
1. 用户点击注册或者找回密码
2. 使用 key + 内容的方式生成 md5 密钥。并发送附带密钥链接的相应邮件
3. 用户收到邮件后点击邮件
4. 验证密钥，处理相应的邮件链接，并执行业务

模块设计图

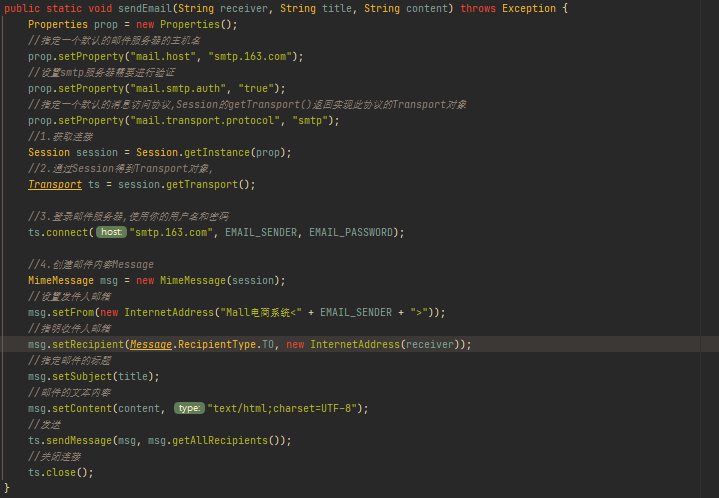


#### 主要代码

邮件工具类



发送邮件主方法

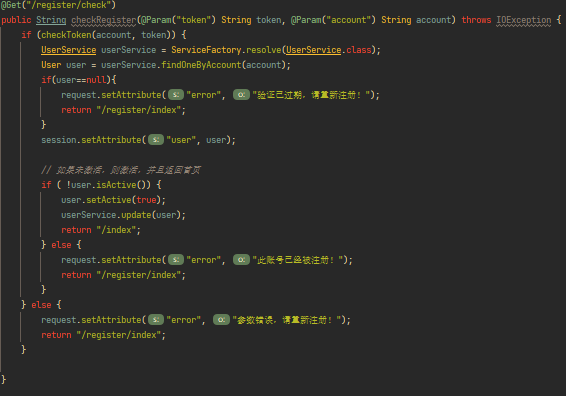


#### 运行截图

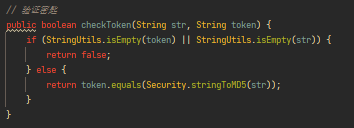
CommonController.java 注册接口



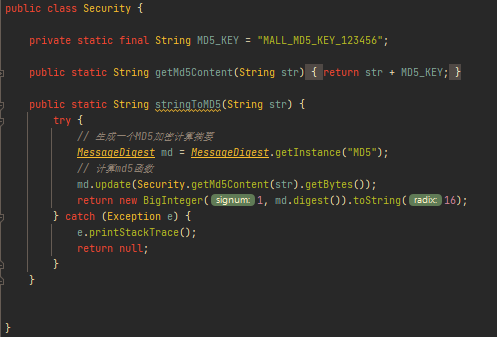
CommonController.java 注册检测



MD5 密钥验证



web安全类实现



注册页面  
 

邮件页面



点击链接验证，注册成功

fig:

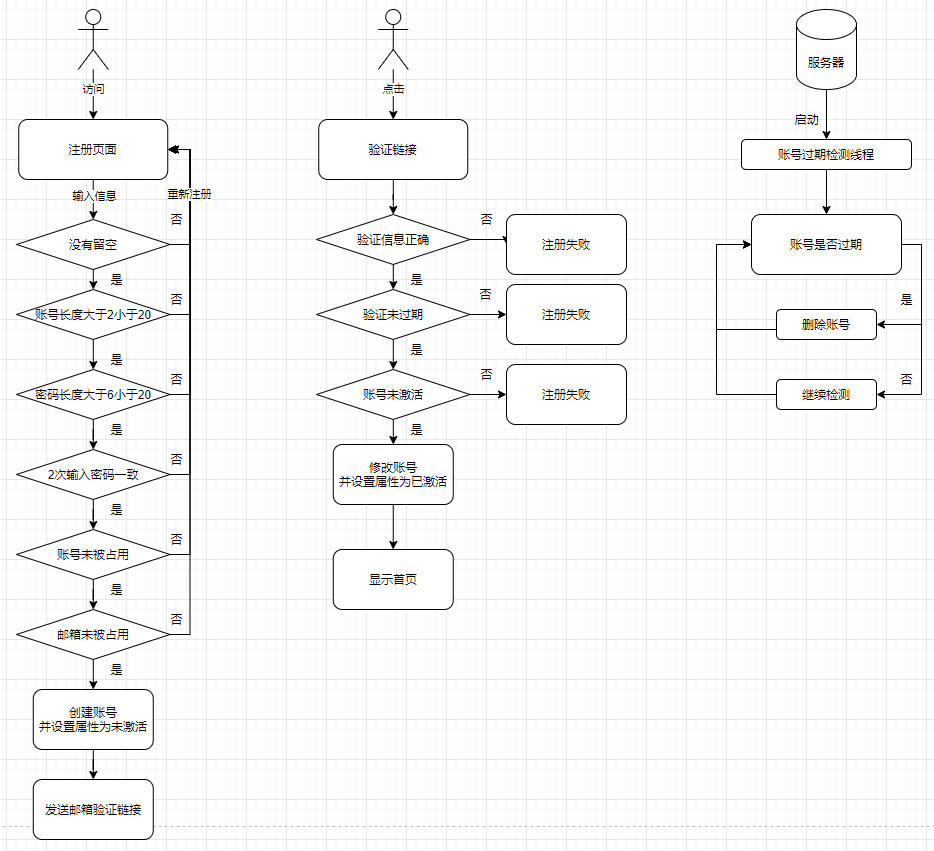
跳转到首页



### 用户模块

#### 1. 注册

##### 流程描述

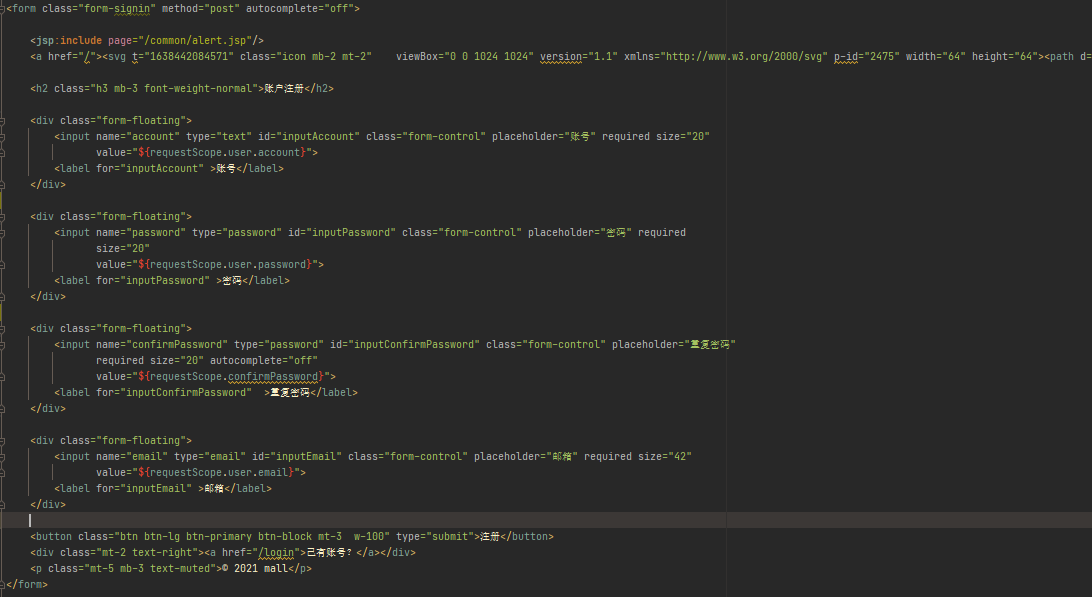


##### 主要代码

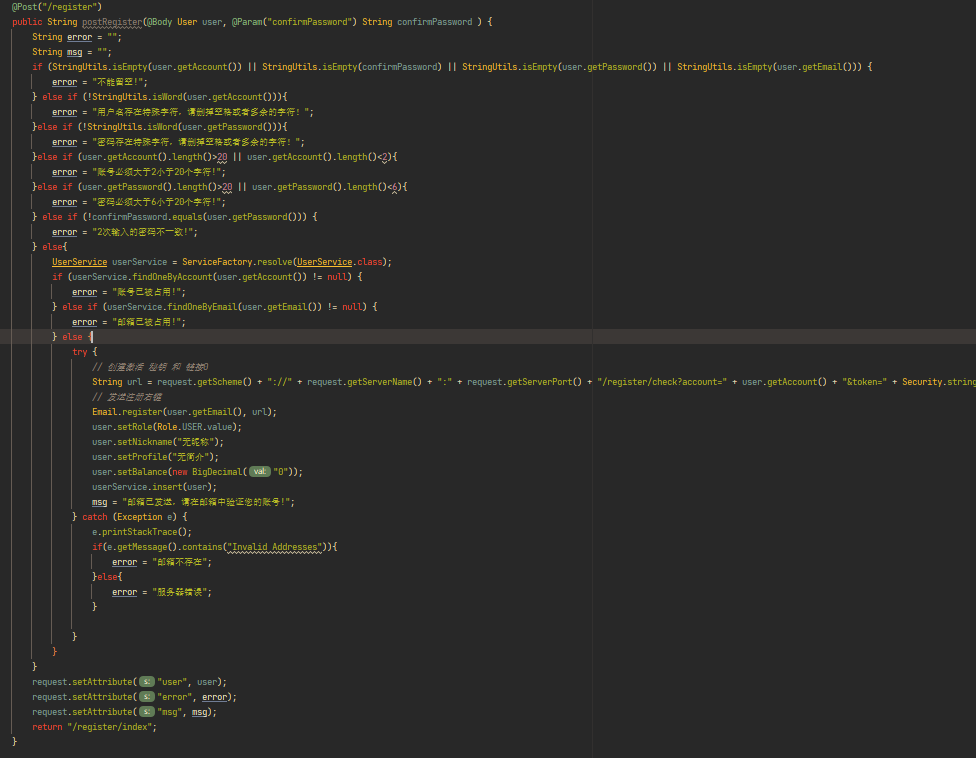
Application.java 账号过期检测



register/index.jsp 注册页面



CommonController.java 注册接口



CommonController.java 注册检测

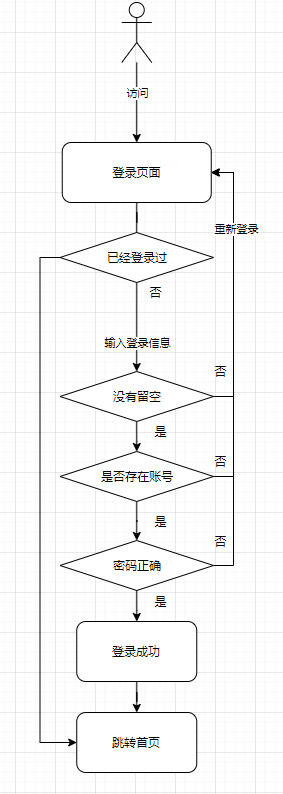


##### 运行截图

参考 [邮件模块](#header-n2728)

#### 2. 登录

##### 流程描述



##### 主要代码

login/index.jsp



CommonController.java 用户登录接口



##### 运行截图

正确的账号密码登录，直接跳转



用户不存在

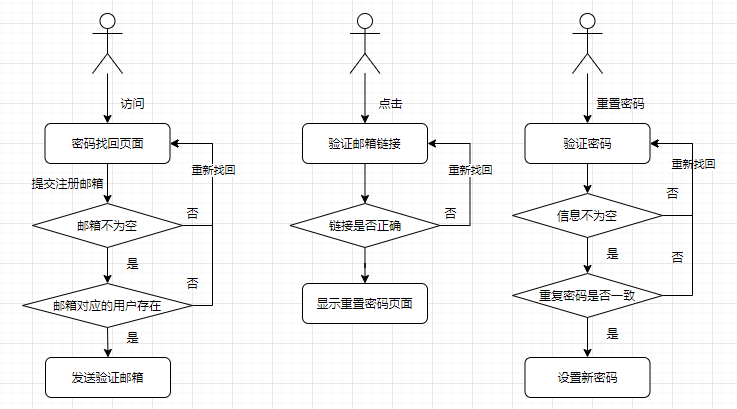


错误的密码



#### 3. 密码找回

##### 流程描述

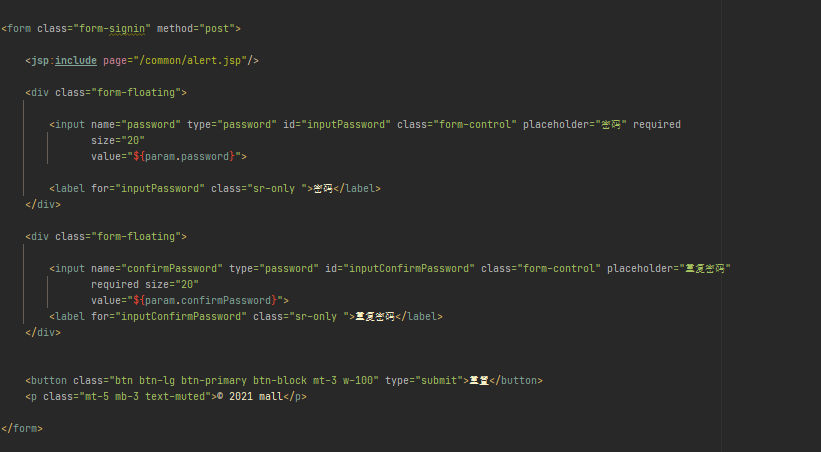


##### 主要代码

forget/index.jsp 密码找回页面



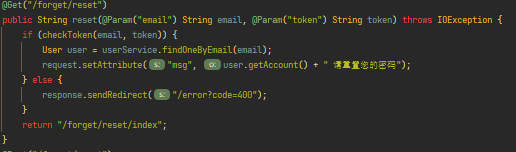
forget/reset/index 重置页面



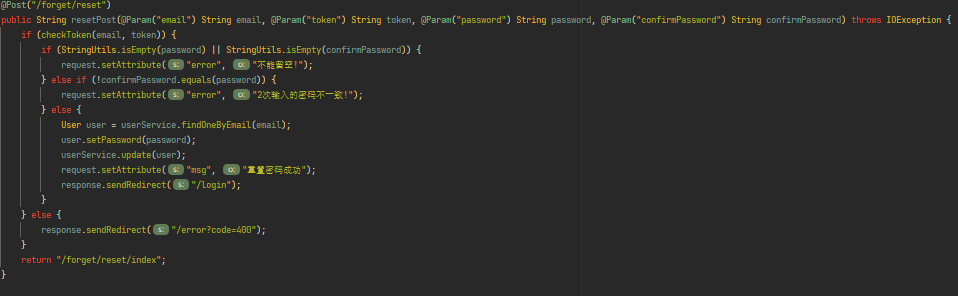
处理忘记密码提交接口



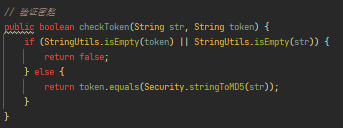
密码重置页面



密码重置页面提交

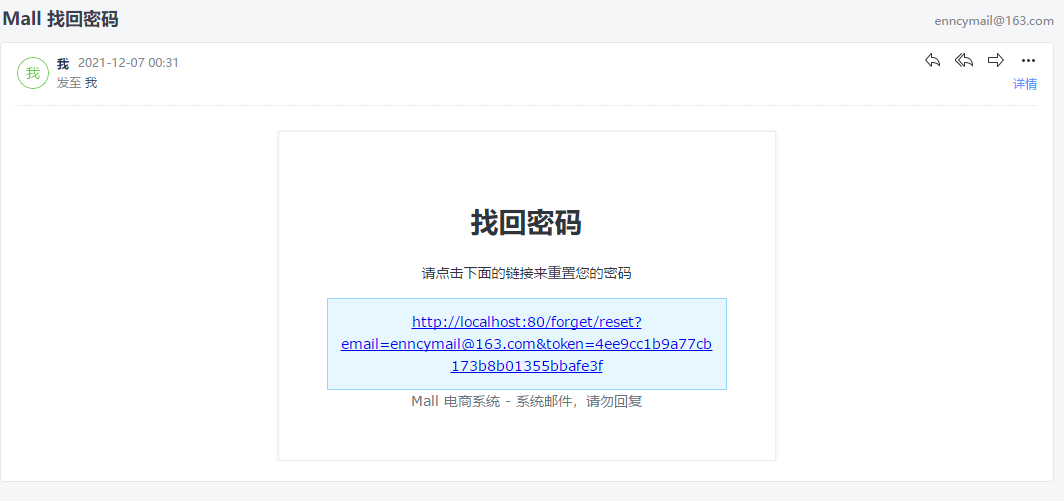


链接密匙验证

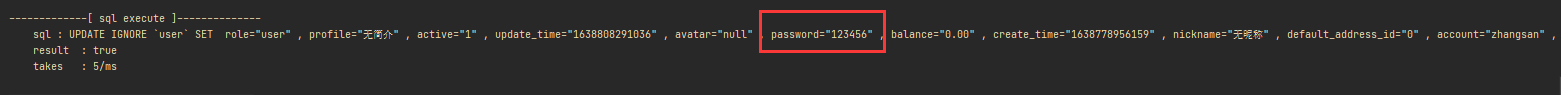


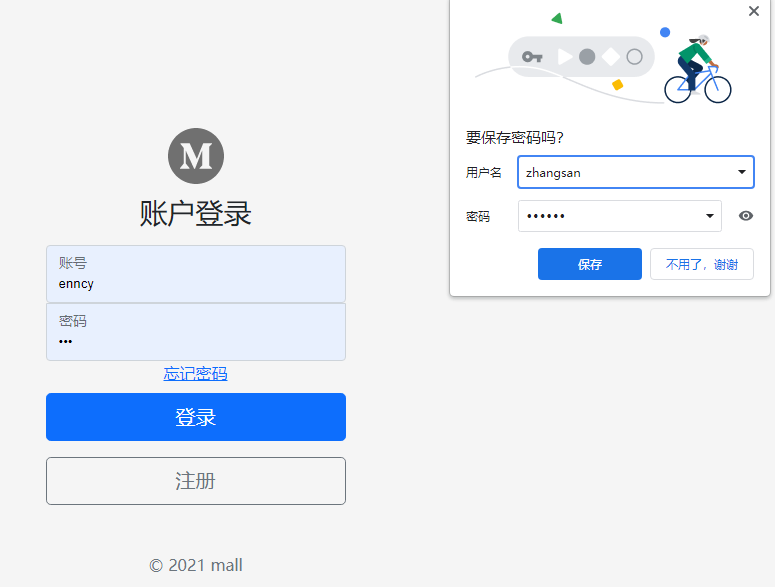
##### 运行截图











#### 4. 用户信息

##### 流程描述

##### 主要代码

##### 运行截图

#### 5. 用户余额

##### 流程描述

##### 主要代码

##### 运行截图

#### 5. 头像修改

##### 流程描述

##### 主要代码

##### 运行截图

前端包括：用户登录，用户注册，用户密码找回，用户信息展示以及修改，用户余额展示

前端后台包括：用户信息增删改查

后端包括：用户登录接口，用户注册接口，用户密码找回接口，邮件发送接口，余额查询修改接口

### 商品模块

### 购物车模块

### 订单模块

## 第五章 设计总结

本次的 java 综合实训 课程设计，让我对 java 有了一个全新的认识，从 java 的封装继承多态中，我学到了如何组织代码中的各个部分，从 java swing 中，我学会了如何管理各种组件，以及组件之间的通讯，传值，调用，依赖，并严格遵守设计模式的6大原则:

1、单一职责原则，实现类要职责单一；

2、里氏替换原则，不要破坏继承体系；

3、依赖倒置原则，要面向接口编程；

4、接口隔离原则，在设计接口的时候要精简单一；

5、迪米特原则，要降低耦合；

6、开闭原则，要对扩展开放，对修改关闭。

实现了 java 的代码复用，实现了低内聚，高耦合的设计模式。并且我对于 java 反射原理也更加进一步的理解并实现， 反射机制其实在很多语言之中都有出现，而反射机制其实简单的来说，其实就是程序在运行的过程中，可以获取到程序本身。

通过 java的反射机制 + 注解 + 动态代理，我自己手动模拟并实现了一个简单的 Mybatis 数据持久层框架，

通过 java 的反射机制 + 注解 + 包扫描，又实现了一个依赖注入原理类似的数据库表初始化系统。

Java 的反射机制是动态程序的关键因素，由此应运而生了许许多多的优秀框架， 从我们最开始接触的Tomcat + Servlet，到现在最流行的 spring, springmvc, springboot,springcloud 等等等等。不计其数的优秀框架开创了 java 的动态技术的先河。

这次的综合实训让我受益匪浅，希望在以后的实训中我能学到更多！

**参考文献**

[1] (美)Bruce Eckel.Java编程思想[M].机械工业出版社,2007

[2] 龚炳江;文志诚;高建国.Java程序设计[M].人民邮电出版社,2016

[3] 王昊天,于航,商贝宁.Java反射机制概述[J].电子世界,2020

[4] 张丹丹.浅析Java swing组件窗体设计[J].电脑知识与技术,2020

[5] 兰旭辉,熊家军,邓刚.基于MySQL的应用程序设计[J].计算机工程与设计,2004

5 篇，含 3 篇期刊 2 篇图书