



System Design Report

目录

System Design Report	1
项目概述:	2
技术栈:	2
Spring Boot	2
Thymeleaf	2
MyBatis	2
MySQL.....	3
Bootstrap.....	3
后端架构设计:	3
entity 层	3
dao 层	3
service 层.....	3
controller 层	4
项目类图:	4
功能模块:	4
登录.....	4
注册.....	5
查询所有用户和查询所有合同	5
数据库设计:	6
简介:	6
ER 图:	6
测试方案:	6
总结和展望:	8

项目概述：

Transport 项目是一个物流控制系统的网页应用。

技术栈：

Spring Boot

Spring Boot 是基于 Spring 框架的快速开发框架，可以用来创建独立的、生产级别的 Spring 应用程序。它提供了开箱即用的配置，让开发人员可以快速构建和部署应用程序。

它的优点包括：

1. 简单易用：Spring Boot 提供了默认的配置和依赖，可以轻松地创建和部署应用程序。
2. 快速开发：Spring Boot 提供了很多开箱即用的功能，包括自动配置、内嵌服务器、开发工具等，可以让开发人员更快地开发应用程序。
3. 易于测试：Spring Boot 支持单元测试和集成测试，可以帮助开发人员更好地测试应用程序。
4. 生产级别：Spring Boot 提供了健康检查、指标监控等功能，可以让应用程序更易于管理和维护。

Thymeleaf

Thymeleaf 是一个基于 Java 的模板引擎，可以用于 Web 和非 Web 环境中的模板渲染。

它的优点包括：

1. 简单易学：Thymeleaf 的语法类似于 HTML，易于理解和学习。
2. 可以处理 HTML 和 XML：Thymeleaf 可以处理 HTML 和 XML 等多种模板格式，非常灵活。
3. 可以与 Spring 框架集成：Thymeleaf 可以与 Spring 框架集成，可以方便地在 Spring 应用程序中使用。
4. 安全可靠：Thymeleaf 可以防止跨站点脚本攻击（XSS），可以保障应用程序的安全。

MyBatis

MyBatis 是一个持久层框架，它可以将 Java 对象映射到数据库表中，实现 Java 对象与数据库之间的交互。

它的优点包括：

1. 灵活性高：MyBatis 可以根据开发人员的需求进行配置，可以实现灵活的数据库操作。
2. 易于学习：MyBatis 的语法简单明了，易于学习和使用。
3. 可以与 Spring 框架集成：MyBatis 可以与 Spring 框架集成，可以方便地在 Spring 应用程序中使用。

4. 性能优秀：MyBatis 可以实现高效的数据库操作，可以提高应用程序的性能。

MySQL

MySQL 是一个开源的关系型数据库管理系统，支持多种操作系统。

它的优点包括：

1. 简单易用：MySQL 的安装和配置非常简单，易于使用。
2. 可靠性高：MySQL 具有高可靠性和稳定性，可以保证数据的安全。
3. 性能优秀：MySQL 可以实现高效的数据库操作，可以提高应用程序的性能。
4. 易于扩展：MySQL 可以轻松地扩展到多台服务器上，支持高可用性和负载均衡。

Bootstrap

Bootstrap 是一个流行的前端框架，用于快速构建响应式和移动优先的 Web 页面。

它的优点包括：

1. 快速开发：Bootstrap 提供了大量的预定义组件和样式，可以快速构建 Web 页面。
2. 响应式设计：Bootstrap 可以自动适应不同的屏幕尺寸和设备类型，可以为用户提供更好的体验。
3. 浏览器兼容性：Bootstrap 兼容大多数现代浏览器，可以为用户提供一致的体验。
4. 可定制性：Bootstrap 可以根据开发人员的需求进行定制，可以实现个性化的 Web 设计。

后端架构设计：

entity 层

实体层。

主要用于定义与数据库对象应的属性，提供 **get/set** 方法,toString 方法,有参无参构造函数。

dao 层

持久层，主要与数据库交互。

DAO 层可以进行数据业务的处理， Dao 层的数据源和数据库连接的参数都是在配置文件中配置。

service 层

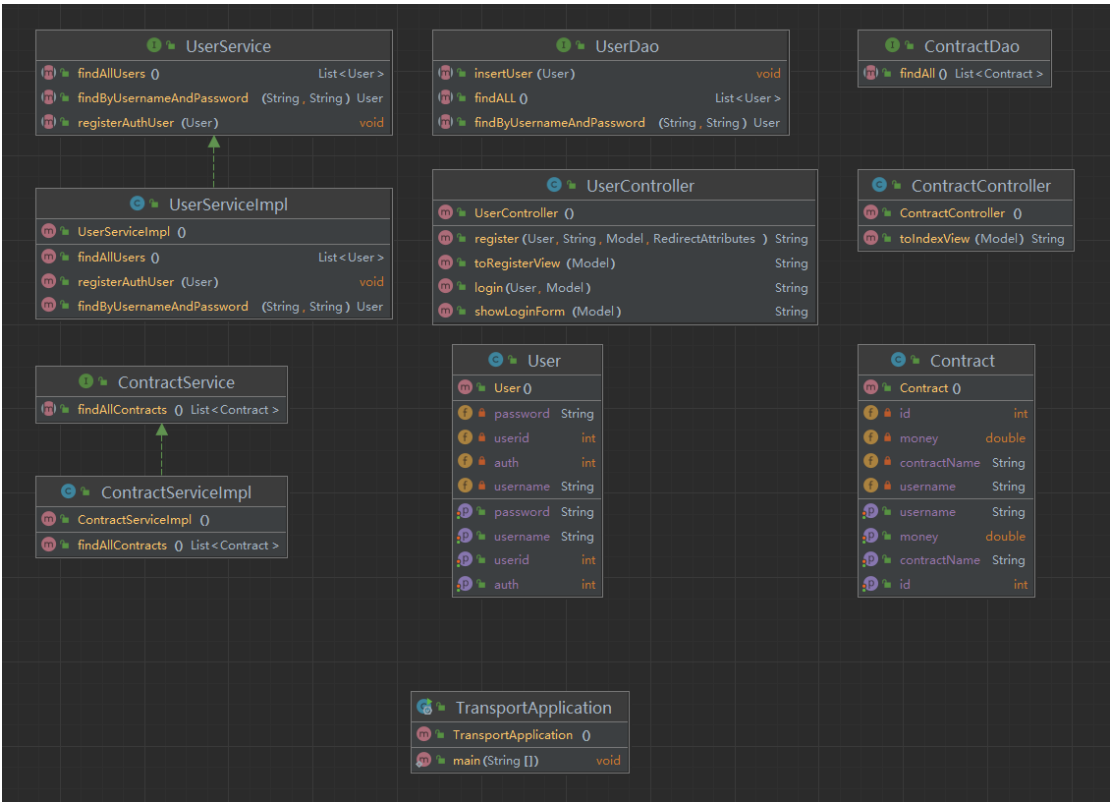
业务层，控制业务。

业务模块的逻辑应用设计，先设计接口，再创建要实现的类，然后在配置文件中配置其实现的关联。接下来就可以在 service 层调用接口进行业务逻辑应用的处理。

controller 层

控制层 控制业务逻辑。
具体的业务模块流程的控制，controller 层主要调用 **Service** 层里面的接口控制具体的业务流程，控制的配置也要在配置文件中配置。

项目类图：

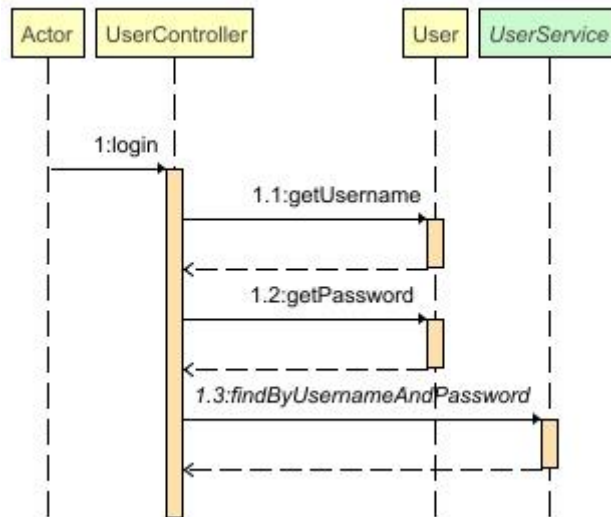


功能模块：

详细介绍每个功能模块，包括其功能需求、实现思路、数据结构等。

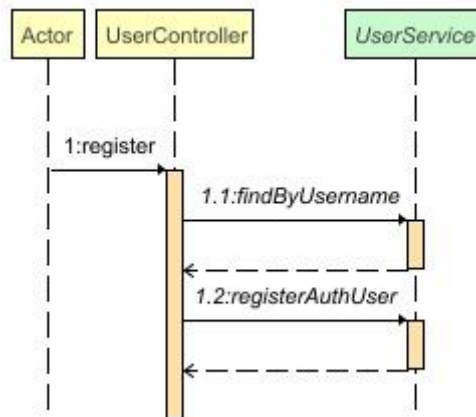
登录

用户输入用户名和密码，进入 UserController 层，UserController 层再调用底层接口查询数据并验证身份。用户登录功能的时序图如图所示：



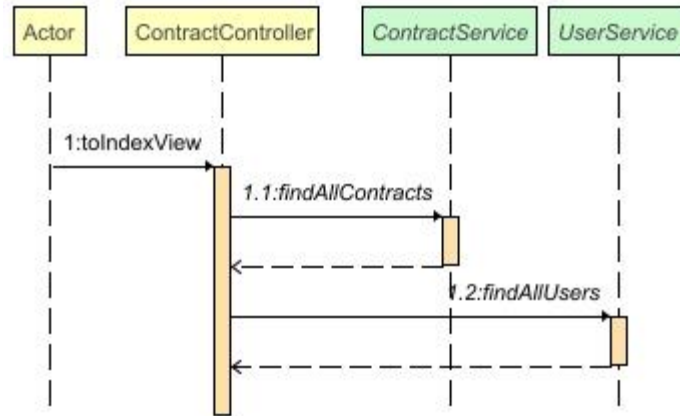
注册

用户输入用户名和密码，以及确认密码后，向 UserController 发送请求，UserController 确认用户名不存在后即向数据库中插入用户信息，并返回登录页面，注册成功。若两次密码不一致或用户已存在则返回注册页面，显示相关错误信息。时序图如图所示：



查询所有用户和查询所有合同

分别向 UserService 和 ContractService 查询用户和合同的所有记录，并返回到主页面展示给用户，时序图如图所示：

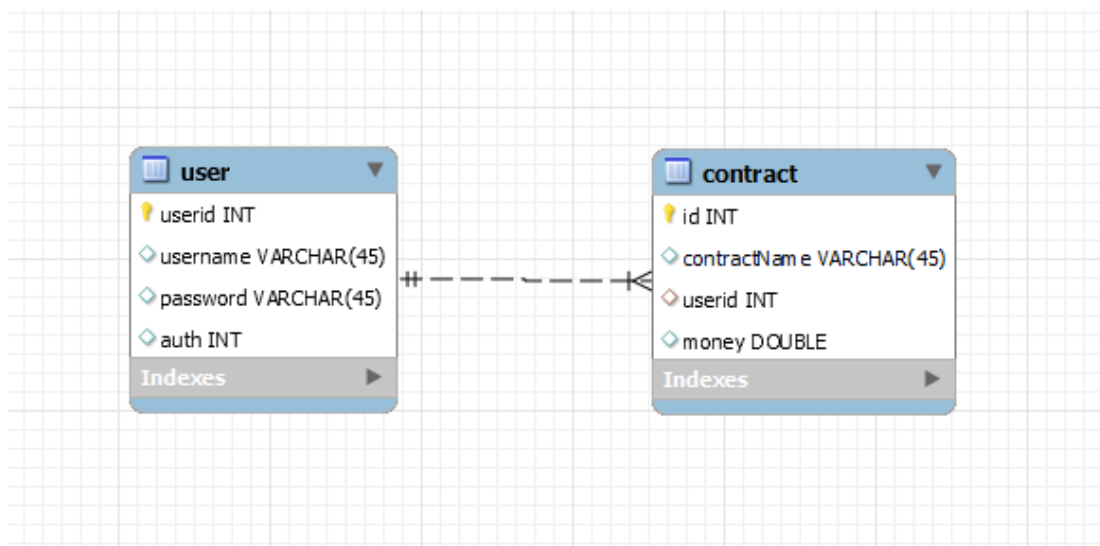


数据库设计：

简介：

数据库目前有 user 和 contract 两张表，其中 contract 表的 userid 属性是外键，关联 user 表中的 userid 主键。ER 图如图所示：

ER 图：



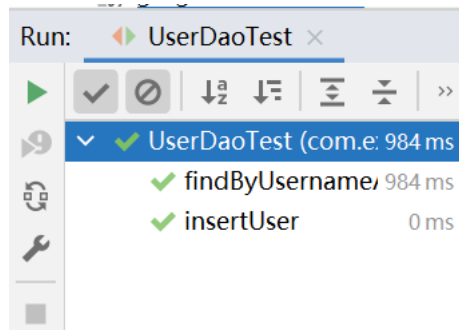
测试方案：

本次测试主要针对 dao 层的各方法进行单元测试和 controller 层的各接口进行集成测试。

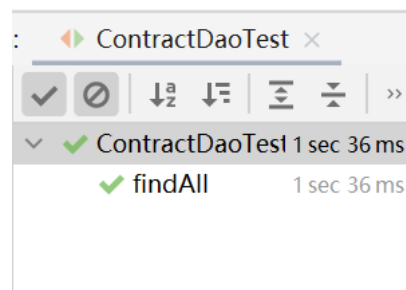
单元测试主要采用 JUnit4 测试框架，添加@Test 注解，在测试方法中使用传统的

System.out.println 对方法结果进行输出查看。

对 UserDao 层的 findByUsernameAndPassword, insertUser 方法进行测试, 测试均通过。

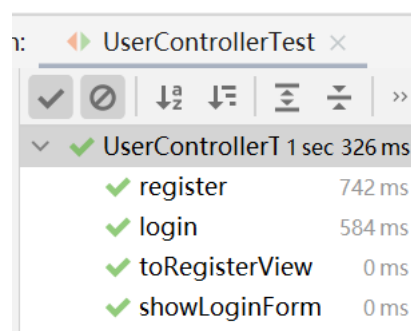


对 ContractDao 层的 findAll 方法进行测试, 测试通过。



集成测试主要采用 MockMvc 工具, 它由 spring-test 包提供, 实现了对 Http 请求的模拟, 能够直接使用网络的形式, 转换到 Controller 的调用, 不依赖网络环境。

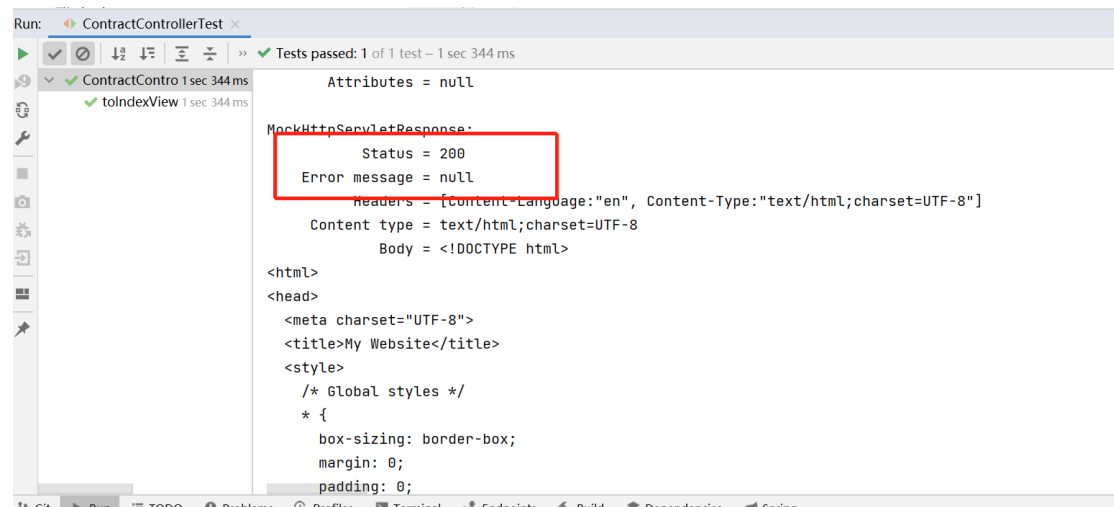
对 UserController 层的 showLoginForm, toRegisterView, login, register 接口进行测试, 测试均通过。



例如 showLoginForm 测试结果如下, 返回状态码为 200, 请求成功:



对 ContractController 层的 toIndexView 接口进行测试，结果如下，返回状态码为 200，请求成功：



```
Run: ContractControllerTest
Tests passed: 1 of 1 test - 1 sec 344 ms

ContractContro 1 sec 344 ms
  toIndexView 1 sec 344 ms

Attributes = null

MockHttpServletRequest:
  Status = 200
  Error message = null
  Headers = [Content-Language:"en", Content-Type:"text/html;charset=UTF-8"]
  Content type = text/html;charset=UTF-8
  Body = <!DOCTYPE html>

<html>
<head>
  <meta charset="UTF-8">
  <title>My Website</title>
  <style>
    /* Global styles */
    * {
      box-sizing: border-box;
      margin: 0;
      padding: 0;
```

总结和展望：

由于初学 thymeleaf，本次作业实现的功能较为简单，是项目的起步。本项目将在后续的 vue+springboot 作业中逐步完善并优化各项功能。本项目的测试部分使用 JUnit4 单元测试框架和 MockMvc 工具进行代码测试，有利于调式代码，发现错误，使得项目的开发更加便捷高效。同时测试框架庞大，在后续作业中需要不断使用新的方法进行完善。