



华南师范大学

本科毕业论文

论文题目：网络教育学院资产管理系
统

指导老师：_____单志龙_____

学生姓名：_____李仲贤_____

学号：_____20172132031_____

学院：_____计算机学院_____

专业：_____网络工程_____

毕业时间：_____2021年7月_____

摘 要

本系统是设计用于学院进行资产采购的流程管理,现有的资产采购流程是全部由人工完成,这使得采购流程的进行以及数据的归档整理效率低下。本系统将原来需要通过线下完成的资产采购申请以及审核等流程改为线上,用户可以在线上通过浏览器访问系统的网站提出采购资产的申请以及进行后续的审核等流程;此外还可以对提出的采购申请进行整理,提高了管理的效率。

资产管理系统使用 B/S 结构,前端需要考虑的是用户的使用体验以及与后端的交互;而后端系统则只需要监听并接受来自前端系统的请求和操作数据库。前端使用 Vue.js^[1]进行开发,实现了数据的双向绑定,用户在操作网页上的元素时能及时给出反应,提升了用户的体验;后端则使用 SpringBoot 开发,简化了 Spring 项目中需要配置的操作,使得后端开发只需要关注与前端的交互逻辑,除此以外的设置都可交由 SpringBoot 自动完成^[2]。

关键词: 资产管理系统; 前后端分离; SpringBoot; Vue.js; MySQL

Abstract

This system is designed for the asset procurement process management of the college. The existing asset procurement process is all completed by manual, which makes the procurement process and data archiving and sorting inefficient. The system has changed the process of asset purchase application and review which need to be completed offline to online. Users can visit the website of the system online to put forward the application of asset purchase and carry out follow-up review and other processes through the browser. In addition, the procurement application can be sorted out to improve the efficiency of management.

The asset management system uses the B/S structure, the front end needs to consider the user's experience and interaction with the back end; The back-end system only needs to listen and accept requests from the front-end system and manipulate the database. The front-end is developed by using Vue.js, which realizes the two-way binding of data. The user can respond in time when operating the elements on the web page, which improves the user experience. The back-end is developed using SpringBoot, which simplifies the configuration operations in the Spring project, so that the backend development only needs to focus on the interaction logic with the front-end. Other Settings can be automatically completed by SpringBoot.

目 录

| | |
|------------------|----|
| 摘 要..... | 2 |
| Abstract..... | 3 |
| 1 绪论..... | 5 |
| 1.1 选题背景..... | 5 |
| 1.2 选题意义..... | 5 |
| 2 需求分析..... | 6 |
| 2.1 功能需求..... | 6 |
| 2.2 安全需求..... | 7 |
| 3 系统总体设计..... | 8 |
| 3.1 系统结构设计..... | 8 |
| 3.2 系统功能设计..... | 9 |
| 3.3 数据库设计..... | 12 |
| 4 系统详细设计..... | 14 |
| 4.1 后端系统设计..... | 14 |
| 4.1.1 权限控制..... | 14 |
| 4.1.2 控制层..... | 15 |
| 4.1.3 服务层..... | 17 |
| 4.1.4 数据访问层..... | 17 |
| 4.1.5 数据库..... | 18 |
| 4.2 前端系统设计..... | 23 |
| 5 系统测试..... | 24 |
| 5.1 白盒测试..... | 24 |
| 5.2 黑盒测试..... | 25 |
| 6 结论与展望..... | 26 |
| 6.1 结论..... | 26 |
| 6.2 展望..... | 26 |
| 参 考 文 献..... | 27 |
| 附 录..... | 28 |
| 致 谢..... | 33 |

1 绪论

1.1 选题背景

原有的资产采购流程完全由人手动完成，申请人需要自己下载申请表格，填写完成后打印出来，然后交由领导进行审核并签名，审核通过后采购的负责人要将申请中的每一项数据填写到另一张表格中，再进行下一步的询价流程。在这个过程中，不仅申请的流程复杂、效率低，而且涉及到的表格众多，稍有不慎丢失了其中的一个文件就会使工作量变得更多。而且，由于是线下完成所有流程，所有申请的整理只能通过人工完成，这样不仅效率低，而且容易丢失资料。

1.2 选题意义

开发这个系统的目的就是简化人员在申请资产采购的操作，以及减轻负责人整理所有申请时的负担。系统将原来线下的申请流程通过线上实现，让用户只需要用浏览器访问网页，并在上面提出申请，填写申请内容并提交，这个申请的提交消息就会发送到对应的领导账号上；而领导只需要登陆账号就可以查看到有哪些人提交的申请在等待审核。这些提交的申请在系统中都会保存下来，并提供给管理员查看，当需要整理所有申请时就可以在系统上找到所有申请的记录。

2 需求分析

2.1 功能需求

通过了解原来线下的申请流程，系统在线上重现了申请流程，并将使用该系统的人分为了四个角色：

普通用户，该角色可以在系统上提出资产采购申请，并且可以查看自己提出的申请的审核进度；

部门领导，部门领导除了可以像普通用户一样提出申请，还可以审核部门中所有已提交的待审核申请单；此外，部门领导也可以查看部门中的所有申请单的进度；

主管院领导，通过了部门领导的审核后的申请就需要等待主管院领导的审核，主管院领导还可以查看系统中的所有申请；

管理员，管理员在系统中拥有最高权限，可以对系统中的用户进行增删改；对于已经通过主管院领导审核的申请，管理员还可以将其中的每一条申请设备按照类型分类，并打包为一份采购单，指派给采购负责人进行采购流程。

按照不同模块分类，系统共有以下模块：

1.**登录模块**：用户的登录和退出。

2.**用户信息模块**：用户可以在此查看个人信息，并且可以修改登录密码。

3.**申请单模块**：普通用户可以在此提交新的申请单、查看已提交申请单的审核进度；部门领导可以查看部门所有的申请，审核部门中的申请，并同时有普通用户的功能；主管院领导可以查看系统中的所有申请，审核已经通过部门领导审核的申请，并同时拥有普通用户的功能；管理员除了可以提交、查看自身的申请外，还可以将申请中的每一项设备分别打包为不同的采购单，分配给采购负责人。

4.采购单模块：此模块只开放给管理员以及拥有采购负责人这一身份的用户，拥有权限的用户可在此查看采购单的详细内容并进行之后的采购流程，然后在系统上更新该采购单的状态。

5.管理模块：管理员可以在用户管理中查看并修改系统中所有用户的详细信息，帮助忘记密码的用户修改密码，手动添加一名新用户或通过 Excel 表格的形式批量导入新用户；此外，管理模块还提供给管理员修改采购经费代码的功能，确保用户在提交申请单时使用的是最新、有效的经费代码。

2.2 安全需求

由于是前后端分离的系统结构，前端向后端发出的请求需要经过验证决定是否放行；系统选用的用户身份验证方式为 JWT^[3]，与传统 session 认证方式相比，JWT 让服务器不需要记住已经登陆的用户，而是通过验证每次请求中携带的认证信息判断该请求的用户身份，以此决定是否通过此次请求。这样既保证了系统的权限安全，也减轻了服务器的负担。

3 系统总体设计

3.1 系统结构设计

系统总体上分成两个部分，前端浏览器部分主要负责展示页面，提供用户可交互的网页页面，并根据用户操作与后端服务器通信，以及将获取到的数据动态渲染到页面上；后端服务器部分主要负责接收来自前端的请求，验证用户权限，解析请求中的数据，以及操作数据库对数据进行增删查改。

前端使用 Vue.js 框架搭建，该框架实现了**数据的双向绑定**，可以很方便地从浏览器页面上的元素获取数据并以此发送请求、以及将后端返回的数据动态渲染到页面上；在与后端通信的方式上，前端使用了 axios 发起通信，实现了请求和响应的拦截，能够在发送请求前在其中添加验证字段，并且通过响应中的数据判断请求是否成功；此外，前端还使用了 Vue Router 进行页面的跳转，Vuex 进行状态管理。

后端使用 SpringBoot 搭建，利用 SpringSecurity 验证用户请求的合法性，每次在接收到前端的请求时，系统会通过请求中携带的验证信息，检查用户角色身份，判断当前用户是否有权限请求该数据；如果验证信息过期或用户角色权限不足，则会拒绝请求并返回失败信息。数据库操作层面，系统使用 JPA^[4]实现从数据库表到 Java 实体类的映射，即 ORM 方案；JPA 提供了一套简单的编程模型，能够非常简单的建立数据表到 Java 实体类的映射，而且 JPA 提供的操作数据库的接口十分简单，需要对数据进行增删查改时，只需要通过拼接 JPA 提供的关键词以及建立映射的实体类，传入参数后 JPA 会自动生成对应的 SQL 语句并操作数据库。

3.2 系统功能设计

系统从角色以及功能上分类，共可分为 10 个模块

(1) **用户模块**：用户的登录和退出，查看用户信息，以及修改用户密码。用户访问前端登录页，输入账号密码后点击登录，前端会将用户的账号密码通过 POST 请求发送给后端服务器，服务器通过拦截器拦下登录请求，交由 Service 层验证用户账号和密码是否正确，验证通过后调用 JWT 相关方法生成有效期为 1 小时的 token，并将其放在响应的头部；退出时，服务器会将签发的对应用户的 token 销毁，系统会拒绝携带已销毁的 token 的请求。用户需要修改密码时，需要在前端输入旧密码和两次新密码，然后前端将新旧密码发送到后端，交由后端验证用户旧密码的正确性；成功修改密码后，后端会将签发的 token 销毁，且前端会在成功修改密码后自动跳转到登录页。

(2) **菜单模块**：在用户成功登录时，后端返回的成功消息中会携带菜单信息，前端通过这个菜单信息，动态渲染出页面上的菜单，这样可以确保不同的用户角色能够看到相应的菜单。

(3) **申请单模块**：这个模块是用户对申请单进行增删查改的模块，主要的功能有新增申请、更新申请、撤销提交申请、获取用户申请单列表、查询一条申请、打印申请单、上传和下载文件、下载可行性报告文件；其中，打印申请单能够按照一定格式将选择的申请单输出为 Excel 文件，供用户下载，用户可以通过 Excel 文件自行打印申请单，交给领导审核签名，最终的签名文件还可以通过上传功能，保存到数据库中。

(4) **部门领导模块**：该模块只有角色为部门领导的用户才能看到相应的菜单。其主要的功能有搜索并查看部门所有的申请单、查看待审核的申请单，并选择通过或是打回；部门领导可查看到待审核的申请的详细信息，并以此决定是否通过申请，如果需要打回申请，则还需填写打回原因。

(5) **主管院领导模块**：与部门领导模块相似，主管院领导模块的区别是能够看到系统中所有的申请单。

用户信息

帐号: 12345678911

用户名: 办公室用户1

所属部门: 办公室

是否采购负责人 否

旧密码

新密码

确认新密码

确认修改

图 3-1 查看用户信息页面

申请单id:

申请部门:

请选择

申请日期:

选择日期

 ~

选择日期

申请经费代码:

申请人:

状态:

请选择

重置

搜索

| 编号 | 申请部门 | 申请人 | 申请日期 | 申请经费代码 | 总金额 | 状态 | 操作 |
|------|------|-----|------|--------|-----|----|----|
| 暂无数据 | | | | | | | |

<

1

>

图 3-2 申请单查询条件

（6）**管理员模块：**管理员除了拥有普通用户的权限外，还能对申请单以及用户进行操作。若普通用户需要删除一条申请，则需要联系管理员，管理员可以通过申请单管理功能删除对应的申请；同样的，管理员也可以恢复已删除的申请。在

用户管理功能中，管理员可以对用户信息进行修改，包括密码，同时还可以将用户任命为采购负责人。此外，管理员在添加用户时，可以选择一条一条地增加，或者下载一个用户导入的 Excel 模板，在其中添加多条用户数据，上传到服务器后系统会自动读取其中的用户信息，并增加到数据库中。除了对申请单和用户的管理，管理员还可以修改采购经费代码，设置好的采购经费代码会在用户新建申请时自动填入。

Q 请输入用户id

搜索

添加用户

批量添加用户

| 帐号 | 用户名 | 部门 | 职位 | 是否采购负责人 | 操作 |
|-------------|--------|-----|-------|---------|--------------------|
| 12345678920 | 张三 | 技术部 | 普通用户 | 是 | 编辑 |
| 12345678917 | 技术部领导 | 技术部 | 部门领导 | 否 | 编辑 |
| 12345678916 | 技术部用户 | 技术部 | 普通用户 | 否 | 编辑 |
| 12345678915 | 主管院领导 | 技术部 | 主管院领导 | 否 | 编辑 |
| 12345678914 | 办公室领导 | 办公室 | 部门领导 | 否 | 编辑 |
| 12345678913 | 财务部用户1 | 财务部 | 普通用户 | 否 | 编辑 |
| 12345678912 | 办公室用户2 | 办公室 | 普通用户 | 否 | 编辑 |
| 12345678911 | 办公室用户1 | 办公室 | 普通用户 | 否 | 编辑 |

<

1

>

图 3-3 用户管理页面

（7）**消息模块：**系统会自动创建并发送消息给对应的用户；当用户提交了一份新的申请单，用户对应部门的部门领导就会接收到一条新的申请单提交消息。除此之外，当用户提交的申请单状态发生改变（审核通过或被打回），提出申请的用户就会收到一条新消息。

（8）**设备列表模块：**用户提交的申请中，有时会包含多条申请设备，而这些设

备的类型可能是不同的，但是其他用户提交的申请中，有可能会包含相同类型的设备，这个模块让管理员能够直接查看所有申请中的设备，并通过设备类型进行分类，打包成为一张采购单，管理员还可以指定一个人为该采购单的采购负责人，拥有该角色的用户能够在采购单模块中找到负责的采购单。

（9）**采购单模块**：这个模块只开放给管理员和采购负责人，采购负责人可以在此更新采购单的状态，以便申请人查看申请中设备的采购进度。

（10）**自助查询模块**：为了方便用户新增申请的操作，系统在申请单中的部分字段实现了自动回填功能，当用户新增一条申请时，前端会自动发送请求查询该用户的姓名、所在部门、所有部门信息、采购经费代码，并自动填入相应的区域；此外，在填写人员姓名信息的区域，系统实现了模糊搜索用户输入的姓名，并列出符合条件的人员。

3.3 数据库设计

由于系统在数据库方面给使用了 JPA 实现的 ORM 框架，所以系统中的每一个实体都对应了一个数据库中的表；系统设计的的实体有：用户、账号、角色、部门、申请单、设备列表、菜单、配置、采购单、通知；各个表的具体设计如下，其中实体之间会有不同的对应关系，例如一个用户只属于一个部门，则在用户实体中会存有部门实体的信息，而部门中会有部门内的用户实体列表。

用户：用户名，账号实体，部门实体，通知实体列表；

账号：用户密码，是否可用标记、角色实体、用户实体；

角色：角色名、角色描述、账号实体列表、菜单实体列表；

部门：部门名称、用户实体列表；

申请单：申请部门、申请人、采购经费代码、申请日期、总金额、申请单状态、设备列表实体列表、打回原因、签名文件；

设备列表：物资名称、品牌型号、配置或技术参数、单位、数量、预算单价、预算总价、申请原因、新设备使用人、设备列表状态、申请单实体、采购单实体；

菜单：菜单路径、菜单名称、角色实体、子菜单实体列表；

配置：配置描述、配置值；

采购单：采购单所属用户，采购单状态、创建时间、更新时间、设备列表
实体列表；

通知：通知标题、通知内容、通知创建时间、通知对应申请单、用户实体；

下图为所有实体间的对应关系（主要对应关系有一对一、一对多和多对多）

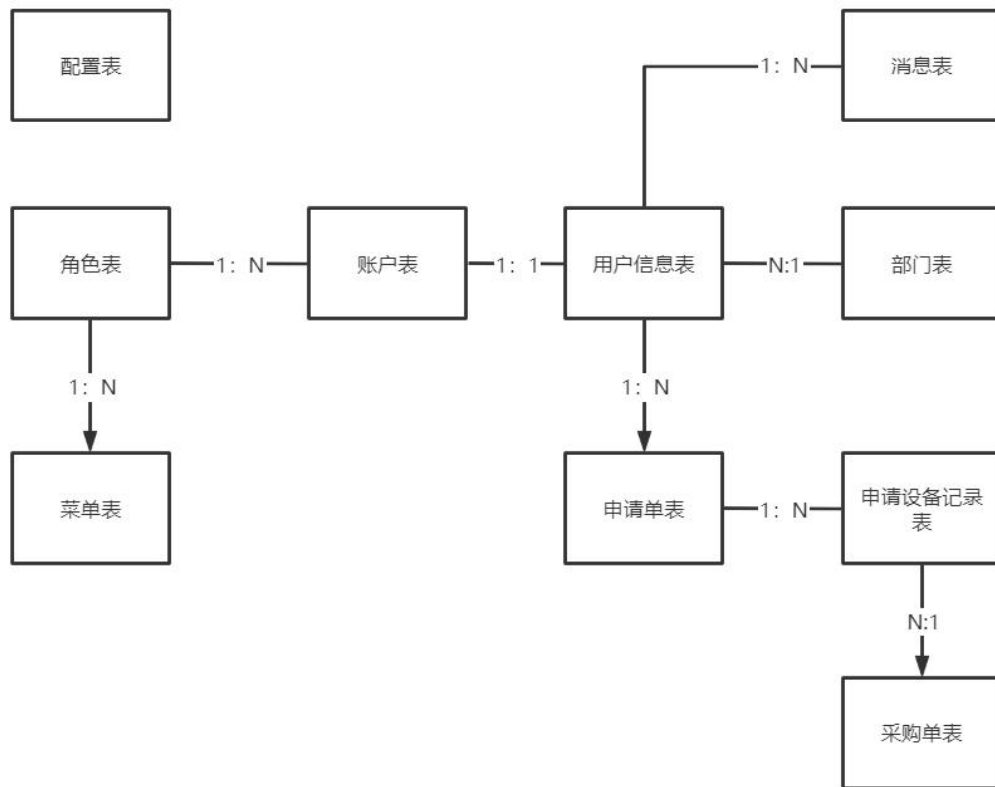


图 3-4 实体关系图

4 系统详细设计

4.1 后端系统设计

4.1.1 权限控制

系统在权限验证上使用了 SpringSecurity 和 JWT。其中 SpringSecurity 主要提供了一组自动配置的过滤器链，通过简单的配置，就能将不同的链接按角色权限区分开来；而 JWT 主要是提供验证用户请求是否合法，后端通过用户在每次请求时携带的 token 验证用户权限，以此决定是否放行该请求。但是，由于 JWT 的特点是无状态，即后端只通过 token 来验证这次请求是否合法，只要 token 还在有效期内，携带该 token 信息的合法请求就能通过验证。

但是，无状态的用户权限验证带来了一个新的问题，那就是当用户在前端点击了退出登录、或者修改了密码后，原先后端签发的 token 只要还在有效期内，那么携带该信息的请求就能被通过，并且后端不能控制对同一个账户签发的 token 的数量，十分不利于系统的安全和稳定；于是，在原来 SpringSecurity 和 JWT 的基础上，引入了缓存机制。在签发 token 时，后端会将由用户 ID 和对应的 token 组成的键值对储存起来，并确保一个用户只保存一个 token；在验证 token 前，系统会先在缓存中通过用户 ID 查找对应 token，然后将请求中的 token 与查找出来的结果作比较，以此决定是否进行下面的流程。SpringBoot 的自动配置中其实有集成了几个常用的缓存，但这些缓存只是用于缓存数据库的查询结果，并不能用来自定义缓存数据，所以，系统引入了 EhCache 缓存，这个缓存将缓存数据保存到内存或磁盘中，通过 xml 文件配置缓存设置。EhCache 与 Redis 缓存相比，Redis 虽然也使用了内存进行缓存，但 EhCache 只需要在项目的 pom 文件中引入 jar 包，就可以做到开箱即用，极大提高了方便程度，而且考虑到使用系统缓存的目的只是为了储存每个登录用户的 ID 和 token，所以最终选择 EhCache 作为缓存工具。

此外，基于 JWT 的验证方式还需要考虑的一个问题就是 token 的有效期。系统签发的 token 有效期是 1 个小时，只要过了有效期限，token 就会失效，这时就会出现用户在前一次请求中还能通过，但下一次请求可能会因为 token 过期而被拒绝。因此，系统在每次验证 token 的有效性时，还会检查 token 的过期时间，如果 token 将要过期，就签发一个新的 token，并通过本次请求的响应返回给前端，前端需要用新的 token 替换原有 token。

4.1.2 控制层

控制层是系统接收前端请求的地方，后端接收到前端发起的请求后，会先通过过滤器验证 token 的合法性，验证成功后过滤器放行，请求到达控制层，然后通过请求 URL 判断调用的方法。

经过验证的用户有访问该接口的权限，但是在接口开始处理流程前，仍需要经过一步验证，目的是防止水平越权的情况出现。水平越权，即虽然用户拥有权限访问该接口，但通过这个接口，用户获取到了并不属于自身的数据；例如，同样是普通用户，若不开启验证，A 用户就能够登陆后通过自己的 token，访问申请单接口，请求 B 用户的申请单数据。所以，系统需要验明用户身份，并判断该用户是否能够读取某项数据。具体的实现方式是：在需要开启验证的接口中，添加一个处理逻辑，通过前端传递的 token，获取用户具体身份，并由此判断该用户是否真正拥有该数据的读写权限。

（1）登录模块

在系统接口中，唯一不需要经过过滤器的接口就是登录接口，而且登录接口的实现并不是在控制层上的，当接收到前端的登录请求，这个请求会进入另一个过滤器中；这个过滤器专门用来接收登录请求，主要的作用就是验证登陆账号的用户名密码是否正确，token 的生成，还有登陆成功后通过角色信息查找并返回菜单数据，供前端动态渲染菜单。

（2）用户信息模块

目前用户信息只有查看个人信息以及修改密码功能。用户在修改密码时，需要输入旧密码以及两次新密码，并检查两次新密码是否相同，防止用户输入错误。

（3）申请单模块

在查询所有申请单接口中，系统还实现了多个字段的模糊搜索功能。接口接收申请单 id、申请日期时间段、申请人、采购经费代码、申请单状态字段；如果传入申请单 id 为“2021”，则系统会返回该用户所有申请单 id 中包含“2021”的申请单，其他字段也是一样的功能；而如果不传递该字段，或传递过来的字段值为空，则不使用该字段进行申请单的筛选。

申请单模块还提供用户下载 Excel 文件形式的申请单的接口，接口接收申请单 id 作为参数，借助服务层找到申请单的详细信息，然后在内存中创建一个 Excel 文件，将数据写入后像前端返回文件流。读写 Excel 文件的工具使用的是 Apache POI, 这是一个用 Java 编写的免费开源的跨平台 Java API, 提供了对 Excel、Word、PowerPoint 等格式文件的读写功能。

（4）采购单模块

在采购单模块中，有一个与系统中其他角色不同的采购负责人角色，这个角色由管理员分配，系统中的所有用户都可以成为该角色，同时原有角色不受影响。采购负责人的角色被记录在配置表中，以用户 ID 作为配置名。由于采购负责人需要在菜单返回数据中添加采购单菜单项，因此在登录过滤器中，系统会查找配置表，判断登录用户是否采购负责人，并在返回菜单中添加采购单菜单。

拥有采购负责人角色的用户能够通过菜单进入采购单管理中，前端会显示一个列表，内容是所有的采购单，采购负责人能够在该页面修改某条采购单的状态，采购单状态更新时，相应申请单中设备列表的状态也会变更，并且后端系统会发送消息至对应用户账号上。

（5）管理模块

管理模块包括用户、申请单、设备列表、采购经费代码的管理。用户管理中，管理员能够批量添加用户，后端提供下载用户批量导入模板以及批量导入用户两个接口；批量导入用户接口需要在前端上传符合格式的 Excel 文件，文件发送到后端，后端系统使用 Apache POI 读取 Excel 文件，从文件第二行开始，每一行中按顺序读取用户 ID、姓名、角色、部门，并将数据存入列表中，直到文件最后一行；之后将该数据传给服务层，服务层会验证数据合法性，并返回每一条数据的操作结果集合，然后，控制层将集合通过相应返回前端，供管理员查看

添加用户结果。需要注意的是，批量导入的用户初始密码是系统预先设定好的值，管理员需要及时向用户告知密码并提醒用户自行修改密码。另外，管理员能够查看并修改系统中所有用户的信息，其中包括用户密码；修改用户信息接口中，系统会判断前端传入的参数字段，并修改对应字段的值，但是系统中用户 ID 的值是标识用户的唯一字段，因此用户 ID 在创建用户时就会确定下来，并且无法改变，也不会出现 ID 重复的情况。

4.1.3 服务层

服务层是系统接口（控制层）和数据库操作对象（数据访问层）的中间层，它能让接收数据和存取数据的操作分离开，让控制层专注于业务逻辑的实现，而无需关注数据操作的问题。这使得控制层不能直接对数据库中的数据做存取操作，增加了系统的安全性。服务层对数据进行校验后，通过调用数据访问层提供的接口实现数据的操作，同时服务层提供相应接口供控制层调用。

总体来说，服务层需要完成对控制层传入的数据的合法性验证，并且在数据不合法时告知控制层，使其向前端返回失败提示。在批量新增用户接口中，服务层的接口接收用户数据列表，对其中的每一条数据，系统会首先查询用户 ID 是否存在，然后检查角色信息及部门信息格式是否正确，在检查出某一个字段的错误后，停止对这条数据的解析，并将出错原因、数据序号等信息存入返回结果集中，然后进行下一条数据的解析；解析成功的数据会首先建立用户和账号两个与数据表对应的实体，并将用户信息存放进去，然后调用数据访问层的接口插入数据，操作成功后记录结果，最后将结果集返回给控制层。

4.1.4 数据访问层

数据访问层是系统中直接操作数据库的地方，在 ORM 的定义中，数据库中的每一张数据表对应一个实体，表中的字段则是实体内的属性。系统使用的 JPA 框架不仅提供了将数据表映射到 Java 实体类中的方法，还提供了通过操作实体类实现对数据库进行增删查改的接口，在数据访问层中，只需要定义好每一个

数据表映射的实体类，然后再实现 JPA 提供的接口，就能够以操作实体类的方式增删查改数据库，无需再关注操作数据库时的 SQL 语句。

由于数据表与实体类是一一对应的关系，数据表中的每个字段都对应着实体中的属性，因此通过实体类的构造，能够清楚地了解数据表的结构。为了避免直接将实体类作为数据传输对象发送给前端，需要建立与实体类相关联的数据传输对象（DTO）。系统对需要隐藏数据表结构对应的实体类上建立了相应的数据传输对象，包括菜单实体、用户实体、申请单实体、设备列表实体；系统在用户数据上使用了两张表来保存数据，而用户 DTO 能同时储存两个实体的数据；用户信息中，用户密码属于敏感信息，于是在查找用户信息，并将信息从实体类储存到 DTO 中时，会先过滤掉用户密码字段，防止用户密码泄露。

在操作数据库时，还需要关注一种情况的发生，那就是并发的问题；如果不同的用户同时对同一条数据进行了改动，就可能会造成数据丢失的情况，即一个用户修改的数据被成功保存，而另一个用户的修改虽然提示了成功，但数据库中并没有保存修改的内容。考虑到该系统的使用是面向一个学院，用户人数较少，产生并发的概率较低，系统使用了乐观锁对数据进行校验，在数据表中添加一个数字字段，该字段用于标明数据的版本信息（修改次数），一条数据发生变化后，该字段的值就会增加。在查找数据时，这个字段也会同样被返回给前端，前端无需对该字段做修改，当前端提交修改过的数据时也需要携带该字段，后端系统通过验证这个字段与数据表中对应数据的字段是否一致，只有字段值相同，这条数据才会被更新。

4.1.5 数据库

系统数据库中数据表的设计如下：

账户表（account）。包括用户 ID，用户密码，账号是否可用标记，角色 ID，openID。其中 open ID 可用于系统与企业微信对接时保存用户企业微信的 ID。

表 4-1 账户表

| 字段名 | 数据类型 | 描述 |
|-----------|-------------|--------|
| id | char(11) | 用户 ID |
| password | varchar(64) | 用户密码 |
| is_active | boolean | 账号是否可用 |
| role_id | int | 角色 ID |
| openid | varchar(64) | openID |

用户信息表（userinfo）。包括用户 ID，用户名，账户 ID，部门 ID。

表 4-2 用户信息表

| 字段名 | 数据类型 | 描述 |
|--------------|-------------|--------|
| id | char(11) | 用户 ID |
| username | varchar(32) | 用户真实姓名 |
| account_id | char(11) | 账户 ID |
| epartment_id | char(2) | 部门 ID |

角色表（role）。角色表中，角色 ID 字段是自增型主键，角色名是系统判断用户身份的字段，存放“USER”，“ADMIN”等字段；而角色描述则是角色中文描述，如普通用户、管理员，是新增用户时管理员需要填写的字段。

表 4-3 角色表

| 字段名 | 数据类型 | 描述 |
|------------------|-------------|-------|
| id | int | 角色 ID |
| role_name | varchar(32) | 角色名 |
| role_description | varchar(32) | 角色描述 |

部门表（department）。

表 4-4 部门表

| 字段名 | 数据类型 | 描述 |
|-----------|-------------|-------|
| id | cahr(2) | 部门 ID |
| dept_name | varchar(32) | 部门名称 |

菜单表（menu）。菜单表中的父菜单 ID 对应的是上级菜单的 ID。菜单的设计分为两层：负责菜单分类的父菜单，该菜单起到将不同菜单分类的作用，只有菜单名称；子菜单不仅有菜单名称，还保存了菜单路径，菜单路径是前端点击菜单后跳转的页面路径。除了与采购单有关的菜单，所有菜单都存储了所属角色的 ID，以便系统根据角色查询对应菜单。

表 4-5 菜单表

| 字段名 | 数据类型 | 描述 |
|------|-------------|--------|
| id | int | 菜单 ID |
| path | varchar(64) | 菜单路径 |
| name | varchar(32) | 菜单名 |
| pid | int | 父菜单 ID |
| rid | int | 角色 ID |

消息表（notice）。消息表中保存了所有用户的消息，表中有一个字段标注该消息是否已读。此外，消息表中还有一个字段存放消息对应申请单 ID，根据此 ID，前端能够在用户点击消息详情时跳转到对应申请单的申请单详情页面中。

表 4-6 消息表

| 字段名 | 数据类型 | 描述 |
|-----|------|-------|
| id | int | 消息 ID |

| | | |
|---------|--------------|--------|
| title | varchar(255) | 消息标题 |
| content | text | 消息内容 |
| state | boolean | 是否未读 |
| uid | char(11) | 用户 ID |
| oid | char(8) | 申请单 ID |
| time | datetime | 消息创建时间 |

配置表（**settings**）。配置表不仅用来保存自定义的采购经费代码，还用于标识用户是否采购负责人。

表 4-7 配置表

| 字段名 | 数据类型 | 描述 |
|-------------|-------------|-------|
| id | int | 配置 ID |
| description | varchar(64) | 描述 |
| value | varchar(64) | 值 |

申请单（**orderapply**）。其中有关部门领导以及主管院领导是否审批的字段有两个，分别是审批标记以及审批的日期。另外，申请单表中还有两个有数据库自动维护的字段，分别是创建日期和更新日期；当有新数据插入到表中，数据库会自动记录当前操作时间为创建日期，同理，更新时间则是该项数据被更新时自动设置为操作的时间。用户上传的申请单照片直接保存在申请单表中的 **file** 字段，

表 4-8 申请单表

| 字段名 | 数据类型 | 描述 |
|------------------|-------------|--------|
| id | char(8) | 申请单 ID |
| apply_department | varchar(16) | 申请部门 |
| apply_user | varchar(32) | 申请人 |
| fund_code | cahr(6) | 采购经费代码 |

| | | |
|-----------------------|--------------|-----------|
| apply_date | datetime | 申请日期 |
| total | double(16,2) | 采购总金额 |
| dept_leader_sign | tinyint(1) | 部门领导签名状态 |
| dept_leader_sign_date | datetime | 部门领导签名日期 |
| inst_leader_sign | tinyint(1) | 主管院领导签名状态 |
| inst_leader_sign_date | datetime | 主管院领导签名日期 |
| status | tinyint(2) | 申请单状态 |
| file | mediumblob | 签名文件 |
| uid | char(11) | 申请人 ID |
| withdrawal_reason | text | 撤回原因 |
| create_time | timestamp | 创建时间 |
| update_time | timestamp | 更新时间 |

申请设备记录（orderlist）。其中的 opinion 是预留字段。

表 4-9 申请设备记录表

| 字段名 | 数据类型 | 描述 |
|--------------------|--------------|--------------|
| id | cahr(10) | 申请设备记录 ID |
| name | varchar(255) | 物资名称 |
| type | varchar(255) | 品牌型号 |
| configuration | text | 配置或技术参数 |
| unit | cahr(1) | 单位 |
| quantity | int | 数量 |
| budget_unit_price | double(12,2) | 预算单价 |
| budget_total_price | double(14,2) | 预算总价 |
| reason | text | 申购原因及旧设备参数状态 |
| new_user | varchar(32) | 新设备使用人 |
| from_id | cahr(8) | 申请单 ID |

| | | |
|-------------|------------|--------|
| status | tinyint(2) | 设备列表状态 |
| purchase_id | int | 采购单 ID |
| opinion | text | 预留字段 |

采购单（purchase_order）。采购单表结构较简单，主要目的只是将已经与申请单关联的设备列表重新分组。其中采购单也拥有创建时间和更新时间两个数据库自动维护的字段。用户 ID 字段则是采购单所属的采购负责人

表 4-10 采购单表

| 字段名 | 数据类型 | 描述 |
|-------------|------------|--------|
| id | int | 采购单 ID |
| status | tinyint(2) | 采购单状态 |
| uid | char(11) | 用户 ID |
| create_time | timestamp | 创建时间 |
| update_time | timestamp | 更新时间 |

4.2 前端系统设计

前端使用 Vue.js，将每一个页面视为一个组件，页面间的跳转则使用 Vue Router 完成；整个系统为单页面应用，切换页面的速度快，用户体验更好。

为优化用户体验，前端在跳转到特定组件时通过预加载的方式，提前从数据库获取数据。例如，在跳转到新建申请单时，系统会像后端发送请求，获取所有部门数据以及采购经费代码的值，这让用户能够在选择申请部门时使用下拉框选择的方式，避免误输入。

在状态管理上，系统使用了 Vuex，将登陆成功后返回的菜单、token 等数据保存到 Vuex 定义的 store 中。

前端使用 Element-UI^[5]作为组件库，页面使用的元素风格与 Element-UI 提供的组件一致。

5 系统测试

5.1 白盒测试

白盒测试就是在熟悉系统内部的工作过程、逻辑结构的情况下对系统内所有操作验证是否符合设计预期。

通过编写测试用例，对系统后端所有接口进行穷举路径测试，保证测试时所有可执行语句都至少执行一次；然后不断改变测试用例输入，验证从输入到输出时系统是否能准确执行。



图 5-1 系统测试用例

5.2 黑盒测试

黑盒测试即系统功能测试，测试时对于系统内部的运行情况是未知的；黑盒测试在前端页面上进行，通过模拟用户的不同操作、输入，验证系统在应对不同情况时候的稳定性。

在后端系统中，如果产生系统错误，那么抛出的错误会被全局异常处理器接收到，并按异常类型生成不同返回结果返回到前端。

在前端登陆页面，通过模拟用户登陆时的不同情况，测试系统在数据错误时的输出提示：

```
▼ {code: 23, msg: "账号密码错误", success: false}
  code: 23
  msg: "账号密码错误"
  success: false
```

图 5-2 登录功能测试

在前端页面上对每个输入框输入不同的值，手工模拟用户可能出现的误操作，测试系统在遇到这些情况时的反应以及输出的结果。经测试，通过前端对每个输入框初步的格式校验、以及后端在接受到数据时对数据的合法性验证，能够保障大部分错误输入时系统运行的稳定性以及合适的返回值。

图 5-3 登录失败提示

6 结论与展望

6.1 结论

在工作分配上，我负责的工作是系统后端的开发。在系统开发过程中也遇到过很多困难；在如何解决 token 在用户活跃时间内过期这个问题上，最初的设想是让 token 的过期时间延长，但签发后的 token 过期时间已经确定，不能改变，因此最终的解决方法是签发新的 token，让前端配合保存新 token 的值。这些问题能够被发现并解决，小组成员和老师的建议和帮助是必不可少的。

6.2 展望

该系统总体上满足了用户提出申请到进行采购过程的需求，但是系统也还有很多可以改进的地方，例如添加管理员管理部门以及角色的功能。另外，前端系统在页面上样式主体还比较简洁，可以在这基础上添加更多动效，优化用户体验。

参 考 文 献

- [1] Vue.js[EB/OL]. <https://cn.vuejs.org/>, 2021-04-20
- [2] 张峰. 应用 SpringBoot 改变 web 应用开发模式[J]. 科技创新与应用, 2017, 000(023):193-194.
- [3] Dearmadman. 什么是 JWT -- JSON WEB TOKEN[EB/OL].
<https://www.jianshu.com/p/576dbf44b2ae>, 2016-04-22
- [4] John O'Conner. Using the Java Persistence API in Desktop Applications[EB/OL].
<https://www.oracle.com/technical-resources/articles/javase/persistenceapi.html>, 2007-06
- [5] 指南 | Element[EB/OL].
<https://element.eleme.cn/#/zh-CN/guide/design>, 2021-04-20

附 录

生成申请单文件代码:

```
@GetMapping("/file")

public void generateFile(HttpServletResponse response,

                        @RequestParam(value = "id") String id)

{

    OrderApply orderApply = orderApplyService.findOne(id);

    try {

        if(orderApply == null){

            PrintWriter writer = response.getWriter();

            writer.write(JSON.toJSONString(ResultTool.fail("找不到申请单")));

            writer.flush();

            writer.close();

            return;

        }

        FileTools.generateExcel(response, orderApply, true);

    }

    catch(IOException e){

        response.setStatus(404);

        e.printStackTrace();

    }

}
```

新增申请单代码:

```
public String addOrder(OrderApply orderApply) {

    String idPrefix = "", id;

    //id 生成策略: 年份(4) 部门编号(2) 部门该年第 n 份申请(2)
```

```

        Department department =
departmentRepository.findByDeptName(orderApply.getApplyDepartment());

        Calendar now = Calendar.getInstance();

        idPrefix = "" + now.get(Calendar.YEAR) + department.getId();

        id = idPrefix + new
DecimalFormat("00").format(orderApplyRepository.countByIdStartsWith(idPrefix) + 1);

        orderApply.setId(id);

        List<OrderList> orderLists = orderApply.getOrderLists();

        int no = 1;

        for(OrderList orderList : orderLists){

            orderList.setId(id + new DecimalFormat("00").format(no++));

            orderList.setStatus(1);

            orderList.setOrderApply(orderApply);

        }

        orderApply.setOrderLists(orderLists);

        orderApply.setStatus(1);

        orderApply.setCreateTime(new Date());

        orderApply.setUpdateTime(new Date());

        orderApplyRepository.save(orderApply);

        return id;

    }

```

登陆成功后返回数据的生成代码：

```

protected void successfulAuthentication(HttpServletRequest request, HttpServletResponse
response, FilterChain chain, Authentication authResult) throws IOException, ServletException {

    response.setContentType("application/json");

    response.setCharacterEncoding("UTF-8");

    List<Menu> menus = new ArrayList<>();

    Map<String, Object> resultMap = new HashMap<>();

    Role role;

```

```

JWTUser jwtUser = (JWTUser) authResult.getPrincipal();

String token = JWTTokenUtils.createToken(jwtUser.getId(), jwtUser.getAuthorities());

response.setHeader("Access-Control-Expose-Headers",
JWTTokenUtils.TOKEN_HEADER);

response.setHeader(JWTTokenUtils.TOKEN_HEADER,
JWTTokenUtils.TOKEN_PREFIX + token);

PrintWriter printWriter = response.getWriter();

Collection<? extends GrantedAuthority> authorities = authResult.getAuthorities();

for(GrantedAuthority authority : authorities){

    String roleName = authority.getAuthority().replace("ROLE_", "");

    role = roleService.findByName(roleName);

    resultMap.put("role", role.getId());

    switch(role.getId()){

        case 1:

            //管理员

            menus.addAll(roleService.findById(4).getMenus());

            menus.addAll(roleService.findById(1).getMenus());

            break;

        case 2:

            //主管院领导

            menus.addAll(roleService.findById(4).getMenus());

            menus.addAll(roleService.findById(2).getMenus());

            break;

        case 3:

            //部门领导

            menus.addAll(roleService.findById(4).getMenus());

            menus.addAll(roleService.findById(3).getMenus());

            break;

        case 4:

```

```

        //普通用户

        menus.addAll(roleService.findById(4).getMenus());

        break;

    default:

        break;

    }

    Settings settings = settingsService.findByDescription(jwtUser.getId());

    if(settings != null) {

        if (settings.getValue() == "1") {

            menus.add(menuService.findByName("采购单"));

        }

    } else if(role.getId() == 1){

        menus.add(menuService.findByName("采购单"));

    }

}

List<ResultMenu> resultMenus = new ArrayList<>();

for(Menu menu : menus){

    resultMenus.add(new ResultMenu(menu.getName(), menu.getPath(),
menu.getChildren()));

}

resultMap.put("menu", resultMenus);    //菜单

resultMap.put("loginName", jwtUser.getUsername());    //用户名

resultMap.put("deptName", jwtUser.getDeptName());    //用户所在部门名称

printWriter.write(JSON.toJSONString(ResultTool.success(resultMap)));

printWriter.flush();

printWriter.close();

//缓存处理

```

```
cacheService.setCacheValue("token",jwtUser.getId(), token);  
}
```

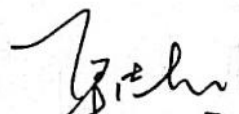
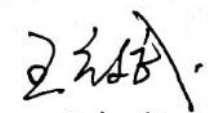
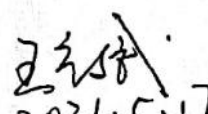

致 谢

该系统以及论文的完成，不是我一人能做到的，十分感谢老师、朋友、小组成员的帮助。

感谢导师单志龙以及网络管理学院的老师卢建晖对我的指导。从论文的选题到平台的构建，再到系统功能逻辑的实现，两位老师都给予了我极大的帮助。

感谢与我一起合作完成该系统的小组成员——罗玉珠同学。明确的分工让我能够专注于后端系统的开发；在讨论时对功能的实现方法上提出了许多建议

华南师范大学本科毕业论文（设计）成绩评定表

| | | | | | |
|-----------------|--------|---|------|---------|--|
| 学生姓名：李仲贤 | | 学号：20172132031 | | 专业：网络工程 | |
| 论文题目：学院资产采购管理系统 | | | | | |
| 指导教师 | 论文得分 | 83.0 | | | |
| | 论文评语 | <p>本项目针对网络教育学院的资产采购问题，开发了一个资产采购管理系统。该系统将原来需要通过线下完成的资产采购申请以及审核等流程改为线上，用户可以在线上通过浏览器访问系统的网站提出采购资产的申请以及进行后续的审核等流程；此外还可以对提出的采购申请进行整理，提高了管理的效率。同意答辩。</p> <p style="text-align: right;">指导教师签名：  时间：2021.5.17</p> | | | |
| 答辩小组 | 是否通过答辩 | 是 | 答辩得分 | 80.0 | |
| | 答辩评语 | <p>该生描述了一种采购系统的设计与实现，能清楚表达项目的开发目标与实现方法。系统基本可用。答辩过程回答问题清晰。通过答辩。</p> <p style="text-align: right;">答辩小组组长：  时间：2021.5.17</p> | | | |
| 最终评分 | | 82.0 | | | |
| 最终评分 | | <p style="text-align: right;">答辩小组组长：  时间：2021.5.17</p> | | | |
| 备注 | | | | | |