



本科毕业设计（论文）

基于web的教室管理系统设计

**学 院 物理与光电工程学院**

**专 业 光电信息科学与工程**

**年级班别 2016级（2）班**

**学 号**  **3116007113**

**学生姓名 陈艺豪**

**指导教师 刘汉瑞**

2020 年 5 月

**基于web的教室管理系统系统设计 陈艺豪 物理与光电工程学院**

**摘要**

随着计算机技术的发展，网络技术和移动互联网技术逐渐成熟，各种管理事务的信息化和自动化应用也越来越多。教室是特别宝贵的资源，特别是在高校中，多媒体教室的设备和环境都是学生学习的最佳选择之一，因此多媒体教室的管理显得尤为重要，因为这关系到教室资源的合理利用。

本论文在对广东工业大学的教室编号进行采集之后，围绕着教室的晚上利用信息，针对广东工业大学多媒体教室的运行现状，运用web（World Wide Web）技术开发了一套教室管理系统。本系统注重于实现学生和管理员都可以查看本周教室的利用信息，如上课、空闲、社团活动等，并分别采用不同页面对教室进行管理。该系统采用前端和后端分离的方式，后端语言采用java语言开发接口，为前端提供数据；前端采用vue.框架+ elementUI实现功能及展示页面。

**关键词：**web技术，教室管理系统，java，vue+elememtUI

**Abstract**

# 目 录

[目 录 4](#_Toc37516353)

[1绪论 6](#_Toc37516354)

[1.1课题的研究背景和研究意义 6](#_Toc37516355)

[1.2 web技术概述 7](#_Toc37516356)

[1.2.1 web技术的概念 7](#_Toc37516357)

[1.2.2嵌入式系统发展现状 7](#_Toc37516358)

[1.3门禁系统概述 8](#_Toc37516359)

[1.3.1门禁系统简介 8](#_Toc37516360)

[1.3.2门禁系统国内外发展现状 10](#_Toc37516361)

[1.4本论文的研究工作和内容 11](#_Toc37516362)

[1.5本论文的内容和结构安排 12](#_Toc37516363)

[2嵌入式门禁系统软硬件方案选取 12](#_Toc37516364)

[2.1系统硬件方案的选取 12](#_Toc37516365)

[2.1.1门禁系统控制器 12](#_Toc37516366)

[2.1.2RFID通信协议读写模块 13](#_Toc37516367)

[2.1.3图形用户界面选择 14](#_Toc37516368)

[2.2门禁系统软件实现的关键技术 14](#_Toc37516369)

[2.2.1系统交叉编译环境的搭建 14](#_Toc37516370)

[2.2.2交叉编译工具链简介 15](#_Toc37516371)

[2.2.3应用程序的移植 15](#_Toc37516372)

[2.2.4Linux下多线程编程 17](#_Toc37516373)

[2.2.5Linux字符设备驱动编程 18](#_Toc37516374)

[2.2.6系统GUI制作 21](#_Toc37516375)

[2.2.7RFID通信协议 23](#_Toc37516376)

[2.2.8Linux系统编程 25](#_Toc37516377)

[3嵌入式门禁系统主控系统软件设计 27](#_Toc37516378)

[3.1系统设备驱动程序 27](#_Toc37516379)

[3.2门禁系统程序编写 32](#_Toc37516380)

[3.2.1门禁系统程序界面构建 33](#_Toc37516381)

[3.2.2界面按键触发操作类 33](#_Toc37516382)

[3.2.3RFID数据采集类 34](#_Toc37516383)

[3.2.4门禁控制线程类 40](#_Toc37516384)

[3.2.5密码信息链表类 46](#_Toc37516385)

[3.2.6非接触卡卡片信息链表类 47](#_Toc37516386)

[3.3门禁系统主程序 48](#_Toc37516387)

[4软件测试与应用 50](#_Toc37516388)

[4.1系统总体测试 50](#_Toc37516389)

[4.2总结 50](#_Toc37516390)

[总结与展望 51](#_Toc37516391)

[参考文献 53](#_Toc37516392)

[致谢 54](#_Toc37516393)

# 1绪论

## 1.1课题的研究背景和研究意义

随着信息时代的到来，当代信息技术的不断发展，网络技术也逐渐成熟，其应用也越来越广泛。近年来，由于web技术的不断发展和成熟，各种功能的网页不断出现，同时也不断改变着我们原有的生活方式，从原本需要一个APP应用才能实现的复杂系统，到现在只需要一个网页或者小程序就能实现其全部功能，这不仅让网页和小程序替代繁琐的APP应用，也大大提高我们的生活效率和办事效率。

在四年的大学生活中，我熟悉到多媒体教室在白天的时候一般用于教学活动，晚上则大多是用于学生自习和社团活动（少数教室用于上课）。白天时有教务系统相应的页面可查看当前教室的信息，这给我们查找空闲教室带来了极大的便利。但是到了晚上，我们只能查到教室的上课信息，却查不到社团活动的信息，因此经常出现在一间空教室学习到一半就有人进来开展社团活动的情况，这无疑给一些热爱学习的人带来极大的困扰；而且社团活动一般比较吵闹，这也给周边上课和自习的教室带来影响。因此，假如对教室资源的利用不规范，会给高校教室使用效率带来严峻的挑战，甚至会让学生彼此产生矛盾进而引发斗殴等更严重的后果。同时作为一个学生，我希望对教室使用情况有一个比较清楚的认识，便于我安排学习或社团等活动，但苦于没有相应的渠道；而对于教师来说，想了解目前教室的安排以及教室的具体使用情况，也没有比较好的解决办法。

基于广东工业大学的基本情况——教学楼数目多，每幢教学楼层数不同，每层楼教室数目也不相同；同时，由于硬件条件的限制，没有更好的地方可供学生进行社团活动，只能占用教室。在当前信息时代的背景下，学生仍然采用跑腿的方式申请社团活动，而后又需经过相关人员进行审批，这对无疑是对学生时间和精力的巨大浪费。当然，教学活动作为教室的主要功能，如何将规模庞大的教室资源合理地分配，做到应用尽用，既不浪费教室资源又不影响彼此进行活动，并且能够提供给学生和教师查询使用，这将是一个比较大的难题。

因此，为了充分利用广工的多媒体教室资源，提高基础设施的管理效率，使教学管理更加信息化、自动化、高效化；同时，既方便教师和学生对当前的多媒体教室情况有一个充分的了解和快速查看，又能提高教室管理者对多媒体教室的管理效率，我设计此套教室管理系统。

## 1.2 web技术概述

### 1.2.1 web技术的概念

Web（World Wide Web）技术，即全球广域网，也称万维网，本意是蜘蛛网，在网页设计中称为网页，表现为三种方式，即超文本（hypertext）、超媒体（hypermedia），超文本传输协议（HTTP），是建立在Internet上的一种网络服务，为浏览者在Internet上查找和浏览信息提供了图形化的、易于访问的直观界面。Web开发技术大体上可分为客户端技术和服务端技术。

1. 客户端技术：

客户端技术也称前端技术，主要任务是展示信息内容，主要包括HTML语言，CSS语言，JS语言。

1. HTML语言：即超文本标记语言，它包括一系列标签，开发者可通过这些标签实现对网页结构的控制，并通过专用的浏览器，将所需要表达的信息按照规则表现出来，是网页的内容构成部分。
2. CSS语言：即层叠样式表，是用来控制网页样式的语言，它通过属性和属性值的方式为网页的样式提供丰富的定义，使网页更加美观，其中包含但不仅限于可以修改网页的位置、背景、字体、边框，甚至还提供了一系列丰富多彩的动画。随着技术的发展，现在已经开发出多套UI可供开发者使用，其中比较常用的有：element-UI、LayUI、MUI等等。
3. JS语言：即JavaScript，它是一种解释性脚本语言（代码不进行预编译），通过API为网页页面添加交互行为，是网页“动起来”的技术支持。JS不同于服务端的的脚本语言，它是在浏览器上运行的，因此不需要服务器的支持；同时它还具有另一个特点：跨平台，这使得它不依赖于操作系统，因此它也受到众多程序员的青睐。而随着技术的发展，JS程序员不满足强大的JS语言只能在客户端使用，因此服务端语言node.js诞生了，这标志着JS同时能在服务端和客户端编写程序；而且，现在发展出许许多多成熟的JS框架，使得代码更加简洁，也使得开发效率大大提高，其中，现在比较流行的有：Vue、Angular、React。这三大框架各有各的特点，所以用于不同的场合。
4. 服务端技术：

服务端也称后端技术，主要为任务是提供信息。主要包括CGI、ASP、PHP、ASP.NET、JSP和Servlet技术。

1. GCI：即公共网关接口技术，是最早用来创建动态网页的技术，它可以使浏览器与服务器之间产生互动。
2. ASP：即活动服务器页面技术，它通过在页面代码中嵌入VBScript或JavaScript脚本语言来生成动态的内容。但是必须在服务端安装适当的解释器，才可以通过调用此解释器来执行脚本程序，然后将执行结果与静态内容部分结合并传送到客户端浏览器上。
3. PHP：即超文本预处理器，它是一种开源的Web服务器脚本语言，与ASP一样可以在页面中加入脚本代码来生成动态内容。对于一些复杂的操作可以封装到类或函数中，在PHP中提供了许多已经定义好的函数。
4. ASP.NET：这种建立动态Web应用程序的技术，是.NET框架的一部分，可以使用任何.NET兼容的语言来编写ASP.NET应用程序。
5. JSP和Servlet：JSP的组合让Java开发者同时拥有了类似CGI程序的集中处理功能和类似PHP的HTML嵌入功能，此外，Java的运行时编译技术也大大提高了Servlet和JSP的执行效率。Servlet和JSP被后来的javaEE平台吸纳为核心技术。

### 1.2.2 web技术的发展现状

随着信息互联网时代的到来，web技术的发展迎来了空前的机遇。现阶段，客户端技术与服务端技术的发展正推动着web技术的快速发展，目前对于web技术的研究和应用的情况如下：

1. 随着web2.0的到来和发展成熟，不仅让web技术更加完善，而且把web技术推向一个更高的高峰，使web技术应用越来越广泛。
2. 框架化是web技术现阶段的重大特点，框架带来的好处数不胜数，它不仅使程序员可以开发动态网页，也使用户有更好的视觉体验。
3. UI框架的发展则是web技术的另一个重要特点，它使得程序员有更多精力去注重逻辑实现，而不需要像以前一样花费很多精力在样式上。
4. 相关语言的发展也为web技术的发展起到了重要作用，HTML5新增了一系列功能强大的语义化标签、CSS3新增的样式属性则丰富了网页样式、ES6（ES为JS的语法标准）中新增的方法以及更加简洁强大的代码风格则受到无数web技术人员的青睐。
5. 前后端的分离使程序员分工明确，更好地各司其职，降低维护成本。
6. 随着互联网技术的参数和web技术的发展，程序web化、应用web化成为必然的趋势。

## 1.3教室管理系统概述

### 1.3.1教室管理系统简介

教室管理系统是专门用来管理多媒体教室的系统，主要是用来管理教室资源，在高校中较常见，包括预约上课、查看教室安排、预约活动等功能。随着现代网络技术的发达，教室管理系统已经由过去的线下人工方式转变为线上智能管理方式，教师学生可根据自己的安排，线上查看或预约教室，这样不仅极大地方便了教师和学生等等用户快速查找空闲教室，也极大地方便了教务处管理人员对教室的管理。一般的教室管理系统应该由页面展示、逻辑实现和数据修改三大部分组成，其中逻辑实现部分除了要实现整个系统的功能之外，还应该注意赋予用户和管理员不同的权限，这也是所有系统在设计时需要注意的一个问题。

根据教室管理系统的发展历史，按照不同的实现方式，可分为以下几类：

1. 传统的线下人工教室管理系统

由于之前网络技术还未发展起来，教室管理只能采用线下人工管理的方式，这种方式不仅浪费人力，而且还不能实时查看和更新教室状态，极大地影响教室的使用，大大地降低了教师和学生的使用体验，同时也使得管理人员的办公效率无法提高。经常出现教师或学生想申请教室，但管理人员不在的情况；或者，教室的状态未能实时更新，而造成一间教室多人使用或者多种活动使用的情况，这无疑给教师和学生以及管理人员带来极大的困扰。故随着网络技术的发达，这种传统的线下的教室管理系统逐渐被舍弃，取而代之的是线上教室管理系统。

1. 基于RBAC的物联网教室管理系统

RBAC，即Role Based Access Control（角色访问控制），采用角色访问控制技术进行权限管理，赋予教室管理人员和普通用户不同级别的权限。该系统根据用户在登陆系统时的实际身份赋予相应的权限，若用户想要更高的权限，可以向拥有更高权限的管理员申请，但必须经过管理员同意之后，才能获得相应的权限，保证了不越权访问。在用户登陆成功之后，会根据不同级别的权限，为用户展示不同的页面，如管理人员登陆之后，可对信息进行审批等操作；而普通用户登陆之后，只能进行查看和申请等操作。

1. 基于ARM+Linux的嵌入式教室管理系统

ARM是对一类微处理器的统称，因为其优越的表现和优良的性能，而被广泛地应用于嵌入式开发。该教室管理系统包括硬件系统和软件系统，其中硬件系统主要包括主控装置、ZigBee协调器和节点硬件模块、RFID阅读器与电子标签、红外遥控器功能学习模块，软件系统主要包括嵌入式Linux操作系统裁剪、交叉开发环境、Qt/Embedded应用程序、嵌入式数据库sqlite和嵌入式服务器。

1. 基于web的教室管理系统

基于web的教室管理系统一般由两部分组成：前端（客户端）和后端（服务端），两部分的分工也很明确，后端负责提供教室和用户的数据，并把数据以接口的方式提供给前端调用。前端通过调用接口发送请求，后端响应请求之后把数据传给前端。前端在拿到后端传来的数据之后可进行一系列操作，登陆时可以对数据进行匹配，用户和密码匹配即登陆成功，进入下一个页面，并把教室的状态以标签的形式渲染到浏览器上，展示出页面。教师和学生可以通过前端页面，发送请求，改变后端数据。这样可以保证实时地更新数据，因为数据是双向绑定的，只要刷新页面就会重新调用接口，而调用接口传来的数据是最新的。

综上所述，以上四种教室管理系统中，传统的线下人工教室管理系统因为耗费巨大的人力物力，并且效率低下，体验感差，因此已基本被淘汰。基于RBAC的物联网教室管理系统虽然能实现基本功能，但是因其技术方面还不够成熟，还有很多方面有待进一步完善，国内外都在努力地研究该方面的知识，相信在不久的将来，在其各方面完善之后，将会广泛应用于国内外大部分学校中。基于ARM+Linux的教室管理系统因为比较复杂，同时涉及硬件和软件方面的知识，虽然在硬件方面可以采用市面上成熟的开发板和开发板模块进行二次开发，但是由于其实用性不太大，同时涉及物联网、嵌入式、以太网及自动控制等方面技术的综合应用，因此要想设计一套成熟的教室管理系统，必将耗费较大的人力物力，所以应用也较少，一般用于教学案例而已。基于web的教室管理系统，因为其轻量级、实用性、高效性等特点，简单易上手，只需一台电脑（服务器）就可以进行开发，因此开发成本也较小，并且现在网络技术的发达也使其安全性有了较好的保障，特别适用于教师和学生等用户的需求，因此被广泛应用。

本设计采用了基于web的教室管理系统设计。可以预见，在未来教室管理系统的发展历程中，更加强大高效的教室管理系统将会进一步得到应用和普及，而效率低、功能少的教室管理系统将逐步被多功能、智能化、自动化的系统所替代。

### 1.3.2国内外教室管理系统发展现状

在国内，现阶段基于互联网开发的教室管理系统已经基本取代了落后的人工教室管理系统，但是由于学校的教室管理系统一般由生产集成商独立研发，其重要目的是实现基本功能即可，一般没有充分了解用户的需求，所以由公司研发出来的教室管理系统，虽说在性能上有一定的优化，但是没办法做到对学校教室有百分之百的了解，因此在细节上也不能百分之百地满足教师和学生的需求。

从其他国家和地区的实际情况来看，由于对传统教室管理系统的漏洞多和教室信息化问题的都有较早认识，以及有强大的网络技术和经济基础做支撑，因此国外比较早教室管理系统的网络化。并且由于起步早，并且不断地推进技术的进步，使得国外一度在教室管理方面遥遥领先国内。

但是近些年来，由于在国内涌现出一大批优秀的开发人员，大大提高了我国在教室管理方面信息化的发展速度。开发出前端框架Vue.js的尤雨溪便是其中之一，Vue.js在近些年可谓是风靡全球，也可以说是web程序员必须掌握的一门技术，它的出现改变了无数网页。正是由于在国内有无数个像尤雨溪这样优秀的开发人员，在细节上不断完善，功能上更加强大，使得多媒体教室管理系统更加多样化。从目前来说，各种网络技术基本都可以实现教室的自动化和实时化管理，其区别只是用不同的手段和方法去实现，应用比较广泛的有基于web的教室管理系统和基于嵌入式的教室管理系统，而作为新技术的物联网，在融合其他技术之后开发出来的教室管理系统也将是一个前景不错的方向。

总而言之，多媒体教室管理的实现是一个不断提升的持续过程。它不仅要随着技术的发展表现出与时俱进的特点，而且要随着用户的需求不断完善其功能。例如，在过去，社团活动等实践活动较少的时候，教室足够举办这些活动，但随着社团活动和其他各种活动的增加，使得原本足够使用的教室呈现出资源不够、影响彼此使用的情况，所以我们需要更智能的、更全面的教室管理系统。因此教室管理系统的提升也是一个借鉴性的提升，我们不仅可以借鉴过去的系统，还可以借鉴别人的系统，并结合自身的需求，开发出最合适的教室管理系统。

## 1.4本论文的研究工作和内容

本文研究工作的目的是为了设计和实现一个实用的教室管理系统网页，实现系统对教室信息的稳定高效管理。

基于web的教室管理系统主要由前端网页和后端数据组成。后端数据的提供是管理系统的核心，也是本次设计的重点和难点，服务器采用Tomcat服务器，语言采用高级语言java编写servlet/JSP页面实现后端接口，系统前端页面通过html、CSS、javascript实现布局并采用vue.js框架和Element-UI组件库优化系统。通过设计，系统能够让用户高效地浏览、查看教室的具体信息，在登陆之后可以租借教室，并且管理员能够处理用户的请求，并将处理结果返回给用户，完成了管理系统对教室的实时查看及高效管理。

本设计主要涉及前后端的交互和计算机网络的内容，本文也主要介绍该教室管理系统的设计及其实现过程，论文的主要工作如下：

1.搭建web网页开发环境；

2.添加Element-UI组件库；

3.前端页面的搭建及功能的实现；

4.后端数据接口的编写；

5.前后端的整合。

## 1.5本论文的内容和结构安排

本设计采用基于Vue.js的设计标准，通过Ajax请求接收后端数据，并将接收的json数据转换成标签渲染到页面中。通过网页可以操作教室管理系统，实现对教室的管理。

本设计的目的如下：

* 掌握前端基础开发知识——HTML、CSS、javascript；
* 掌握前端框架——Vue.js、Element-UI；
* 掌握java基础及servlet/JSP。
* 掌握部分计算机网络知识。

第二章讲述本设计各模块的选取，论证了方案的整体架构以及开发过程中使用的关键技术，最终确定整个系统设计的方案。

第三章是整个教室管理系统设计的灵魂，介绍了每个功能模块的程序设计，包括本管理系统的界面设计及程序的具体实现。

第四章对设计进行了必要的测试，验证设计方案的正确性和实用性。

最后则是对本次毕业设计作了总结和期望。

# 2 基于web教室管理系统各模块的选取

## 2.1系统前端方案的选取

### 2.1.1框架的选取

现在比较流行的框架有：Vue.js、React.js和Angular.js。本管理系统的前端框架采用vue.js框架，其考虑的原因有以下几点：（1）、采用本框架的原因是基于对系统的优化考虑，更好地实现前后端分离；（2）、本人在之前已经自学过该框架，因此首选该框架；（3）、 Vue.js是一种MVVM框架，是一套构建用户界面的渐进式框架，它由于轻便容易上手并且便于和第三方库或既有项目整合。

### 2.1.2 CSS预编译语言的选取

现在比较流行的CSS预编译器有Less和Sass。原始的CSS代码写起来比较杂乱无章，这样不仅影响代码的美观性，而且对修改样式造成很大的麻烦。因此本系统采用Sass对CSS样式进行了优化。

### 2.1.3第三方UI库的选取

使用第三方UI库的原因是为了美化页面以及提高开发效率，而Element-UI就是一个强大的第三方库，相比较之前用过的MUI和Mint-UI，它不仅提供了一整套强大的元素，而且改善了其使用体验。在之前用MUI和Mint-UI时，总会遇到一些莫名其妙的bug，在你没有专门的人士进行指导的时候，很难定位bug出现的原因。在听取同学的建议后，决定在此系统中采用Element-UI库进行开发，而不采用之前用过的MUI和Mint-Ui。

Element-UI的使用步骤如下：

1. 安装：npm i element-ui -S
2. 在main.js中加入Element-UI：

import "./plugins/element.js";

import ElementUI from 'element-ui';

import '../node\_modules/element-ui/lib/theme-chalk/index.css';

Vue.use(ElementUI);

1. 在项目中运用Element-UI的元素

### 打包工具的选取

打包工具可以帮助我们更好地管理项目，为我们的项目锦上添花，并且随着js的发展，出现了一些新的方法，但浏览器却不支持，这时候就需要打包工具为我们转换为浏览器能识别的旧语法。在grunt ， gulp，webpack， rollup四种比较常见的打包工具中，我采用webpack来作为本系统的打包工具，主要基于以下几点：（1）、 webpack的官方文档比较详细，并且更新频率很高；（2）、而且在其他的打包工具在处理非网页文件（比如svg, png, vue等）时基本还是需要借助webpack来实现的，所以最关键现在的脚手架主流依旧是webpack。

Webpack的使用和常见的配置：

1. 安装：cnpm i webpack -g;
2. 执行：webpack 打包文件的路径 输出文件的路径
3. 基本配置：在webpack.config.js中

const path = require('path')

// 在配置文件中，需要手动指定入口和出口

module.exports={

//入口文件，指定webpack要打包的文件

entry:path.join(\_\_dirname,'./src/main.js'),

//出口文件，配置webpack打包生成的文件

output:{

path:path.join(\_\_dirname,'./dist'), //指定出口文件的路径

filename:'bundle.js' //指定出口文件的名称

}

}

在Vue-cli脚手架中，为webpack配置提供了极大的便利，它让使用者不再需要配置繁琐的文件规范，而直接一键生成配置文件。

### 2.1.5 前端编程软件的选择

现在市面上比较流行的前端编程软件有hbuilder和VScode和webstorm，这三者我接触比较多的是VScode，因为我从一开始学习前端用的就是VScode，VScode不仅可以用于编写前端，还支持很多语言，例如java还有数据库SQL语言，具有很好的可扩展性，并且运行起来相比较webstorm较流畅，启动速度也比webstorm快很多。Webstorm之前也有接触过，但是由于它的运行和启动不如VScode，所以渐渐被我舍弃掉。

## 2.2系统后端方案的选取

### 2.2.1 后端语言的选取

本系统可采用的后端语言有很多，如Python、java、node.js等等。在之前学过node.js的前提下，我舍弃node.js而采用java的原因是想利用此次机会学习java的基础语法，并且掌握用java编写接口的方法，然后尝试用jsp/servlet为本系统提供一些数据。我也想通过此次学习，为以后往全栈方向发展打下一个坚实的基础。

### 2.2.2后端编程软件的选择

之前刚开始学习java的时候，为了使真正能学习语法，一直采用记事本的方式编写程序，但是这次编写一个较大工程的系统，编程工具若继续采用记事本，势必会影响效率，而且有些错误也不能及时检查，还有初始化项目的时候异常麻烦，因为每个项目都有很多文件和依赖项。故必须选择一款功能强大的IDE编程软件，作为入门的人来说，我采用了eclipse，我之所以不选择IDEA，是因为电脑内存本身不太够，若选择IDEA作为编写工具，会有死机的风险和运行卡顿的问题。

# 3基于web教室管理系统的程序设计

## 3.1搭建系统

### 3.1.1 前端项目的搭建

本次系统前端部分的搭建，我直接采用VUE-cli脚手架进行搭建，因为Vue-cli脚手架采用选择好配置之后一键生成项目的方式，大大提高了效率，相比较过去采用webpack配置文件的方式，不仅效率提高了，而且省去了繁琐的查资料来配置的过程。

Vue-cli脚手架搭建项目的过程以及所选配置：

1. 在命令行输入vue create classroom-manger
2. 选择配置：自行配置

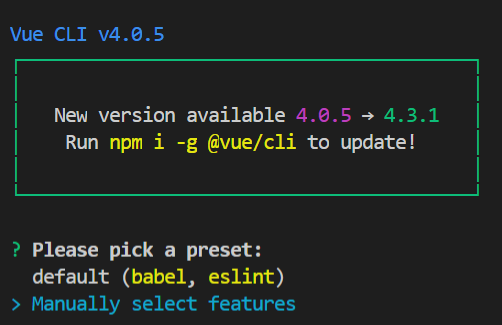


图 2.1 选择自行配置

1. 选择配置：

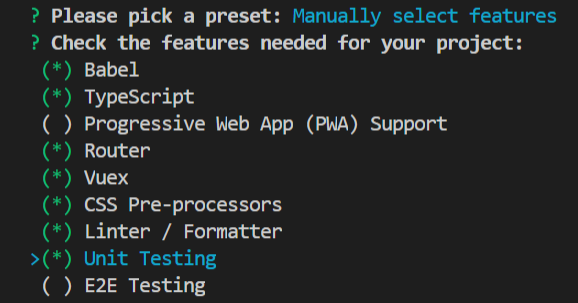


图 1.2 选择配置项

1. 继续选择配置：

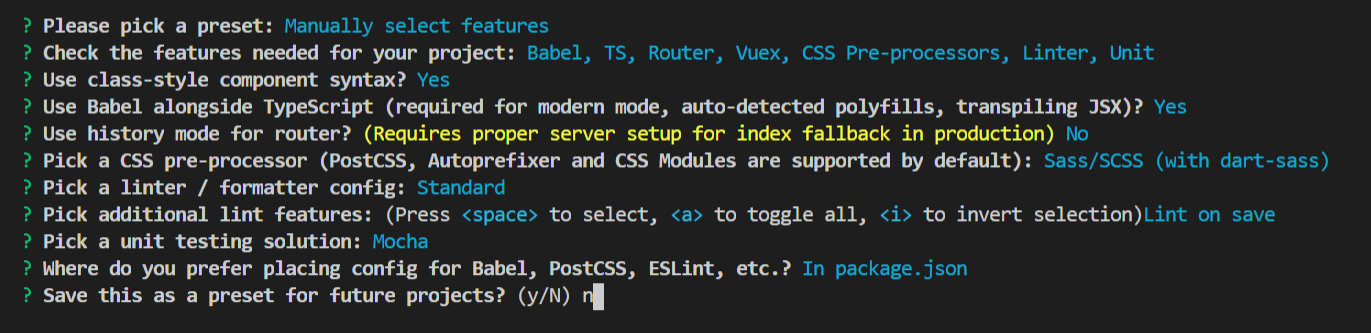


图 2.3 继续选择配置项

1. 进入安装

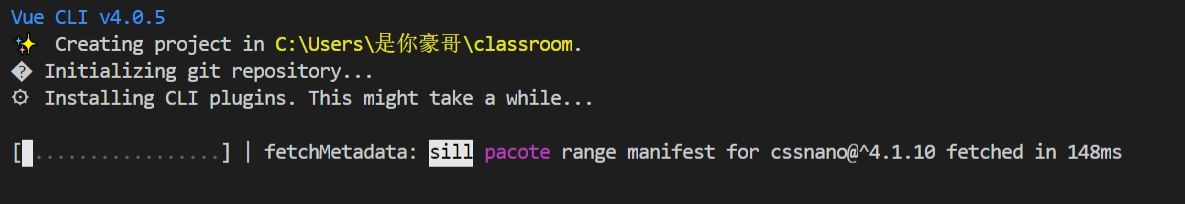


图 2.4 进入安装

1. 安装完成

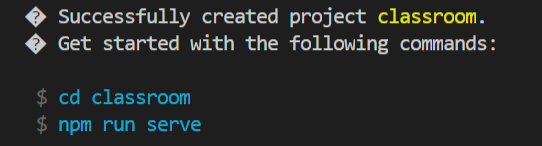


图 2.5 安装完成

1. 测试：进入项目目录并输入npm run serve，并出现以下画面，说明创建成功

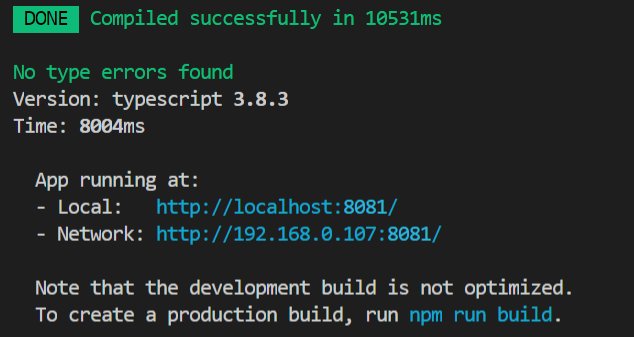


图 2.6 测试

1. Element-UI的引入

|  |
| --- |
| 命令行输入以下命令：  npm install element-ui -save  vue add element  在main.js中引入element：  import ElementUI from 'element-ui';//引入element-ui组件  import 'element-ui/lib/theme-chalk/index.css'; //element-ui组件的样式需要单独引入  Vue.use(ElementUI); //注册element组件 |

1. 项目文件清单

|  |
| --- |
| * node\_modules * public   + favicon.ico   + index.html * src   + assets     - GdutDoor.jpg     - GDUTlogo2.gif     - loginbag.jpg     - timg.png     - user.png   + components     - mangerLogin.vue   + plugins     - element.js   + router     - index.js   + store     - index.js   + views     - classroomMsg.vue     - login.vue   + APP.vue   + element-variables.scss   + main.js * .browserslistrc * .eslintrc.js * .gitignore * babel.config.js * package-lock.json * package.json * README.md |

### 3.1.2 后端项目的搭建

1. 在eclipse中配置Tomcat服务器
2. 新建一个Dynamic Web Project并命名为login（开发学生登陆接口）

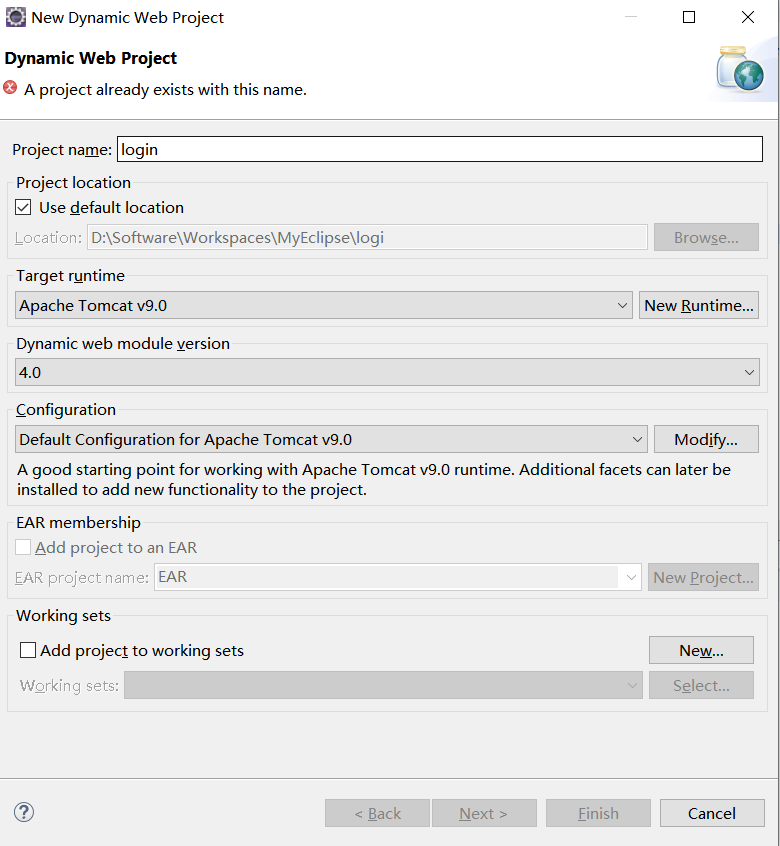


图 2.7 开发学生登陆信息接口

1. 新建完成之后在项目下新建一个servlet项目进行开发，并导入json.jar工具包和Add Build Path，项目文件如下

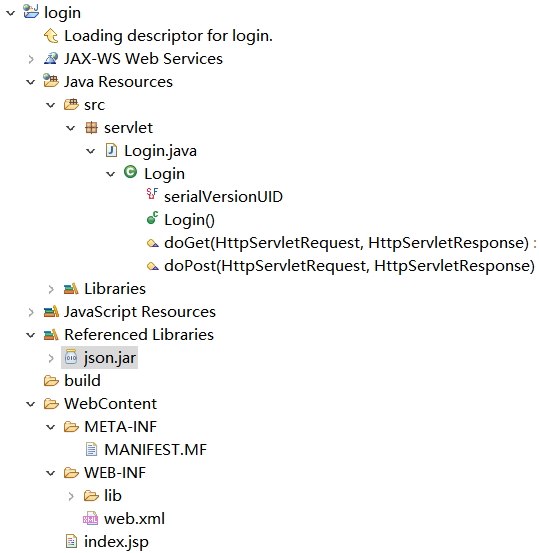


图 2.8 login接口项目文件清单

1. 重复以上步骤，新建一个名为Administrator的项目用以进行管理员信息接口的开发

## 3.2 各项目的关键程序设计

### 3.2.1 前端程序设计

前端程序主要包括主页、学生登陆页面、管理员登陆页面、教室详情页

#### 3.2.1.1 主页的设计

主页主要包含两个跳转链接：查看教室信息和登陆，其分别导向教室详情页和学生登陆界面，并采用背景为淡黄的护眼模式，在底下添加一张广工正门的照片，以说明此系统适用于广东工业大学。



图 3.1 登陆界面视图

其程序各部分设计如下：

|  |
| --- |
| 标签部分：  <div id="app">  <p v-show="flag" class="title">欢迎进入广工教室信息查询系统，请选择</p>  <router-link to="/classroomMsg" v-show="flag">查看教室信息</router-link>  <router-link to="/login" v-show="flag">登陆</router-link>  <router-view />  <img :src="bgcSrc" alt="" v-show="flag">  </div>  数据部分：  flag:true, //用来控制图片是否显示的标志  bgcSrc:require("./assets/GdutDoor.jpg") //图片路径  函数部分：  created() {  this.flag = this.$route.path === "/" ? true : false;  },  watch: {  "$route.path": function(newVal) {  if (newVal === "/") {  this.flag = true;  } else {  this.flag = false;  }  }  },  样式部分：  #app{  display: flex;  align-items: center;  flex-direction: column;  background-color: rgba(255, 166, 0, 0.466);  a{  margin: 3px;  padding: 3px;  text-decoration: none;  color: black;  background-color: skyblue;  border-radius: 2px;  }  img{  width: 80%;  height: 80%;  }  .title{  margin-top: 10px;  }  } |

#### 3.2.1.2 学生/老师登陆界面的设计

当在主页点击登陆链接之后，便进入学生登陆界面，学生登陆界面主要内容是信息输入框和操作选择框，并用一张增加浅色方格图片做背景美化页面和使页面不会过于单调。输入信息框有两个，分别用来监控页面的学号输入和密码输入状况，这两个输入框我采用了v-model的方法将输入框内的内容与程序数据部分进行双向数据绑定，实时更新程序的数据部分。操作选择框主要包括三部分内容，分别为记住密码操作、忘记密码操作和管理员登陆跳转做。记住密码操作由一个复选框构成，用户可以重操作，主要通过cookie存储数据和程序数据两部分联合对数据进行更新和绑定；忘记密码则由于权限问题，只能寻求老师赋予权限，让管理员修改后端接口里边的数据，因此只弹出一个“请联系老师”的提示框；管理员登陆则是跳转到管理员登陆界面，之所以把管理员登陆跳转放到学生的登陆界面里边而不放在主页，是因为此系统主要针对学生和老师，把管理员登陆的跳转链接放到主页会影响主页的美观性和整洁性。



图 3.2 学生/教师登陆界面

该界面对键盘上的enter实行全局监控，并绑定登陆操作事件。本页面的核心程序是对学号/教工号和密码与接收接口的账号信息进行匹配，按下登陆键或键盘按下enter键之后，共有四种匹配状态，分别为：

1. 学号/教工号输入框的账号不存在

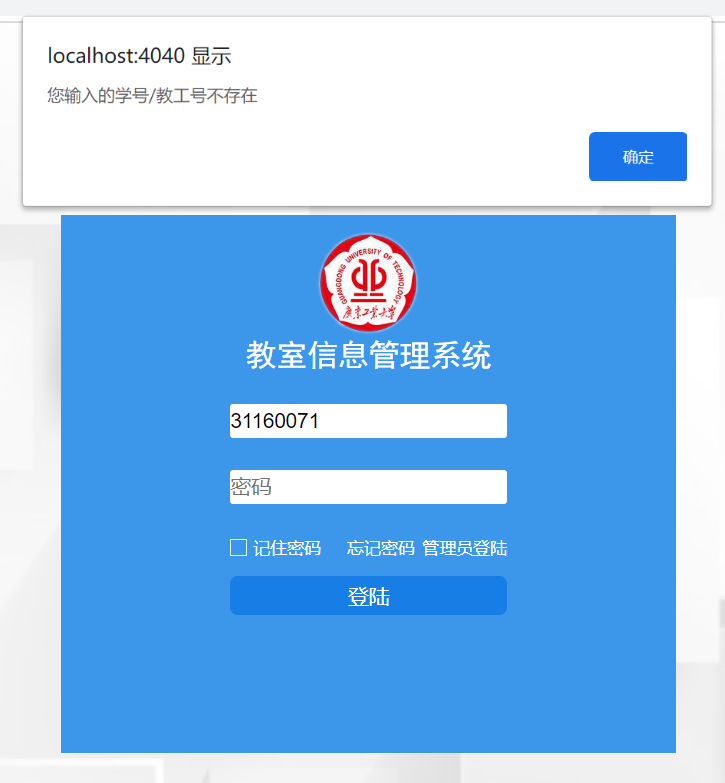


图 3.3 账号不存在

1. 学号/教工号输入框的账号存在，密码为空

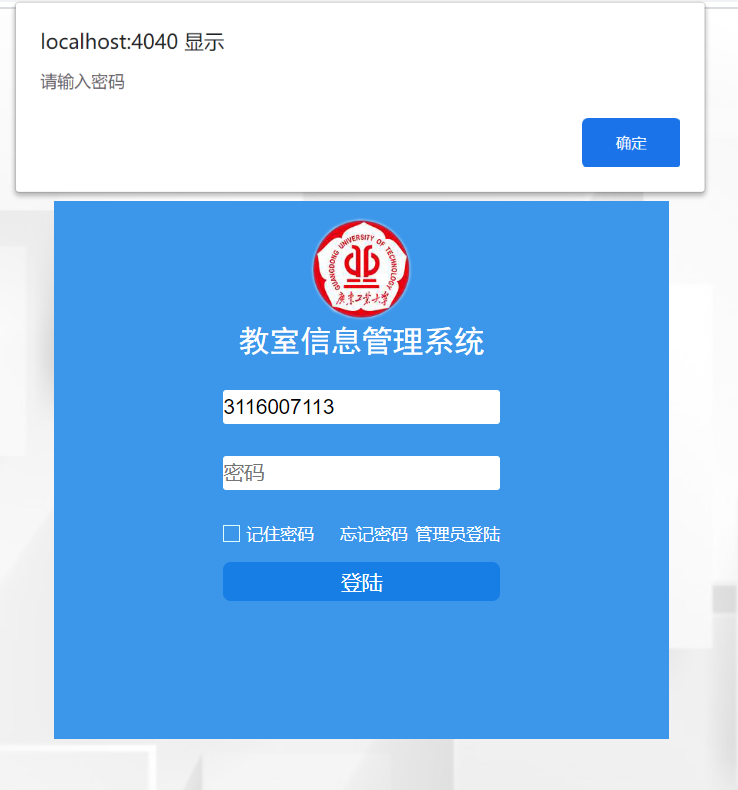


图 3.4 请输入密码

1. 学号/教工号输入框的账号存在，密码错误

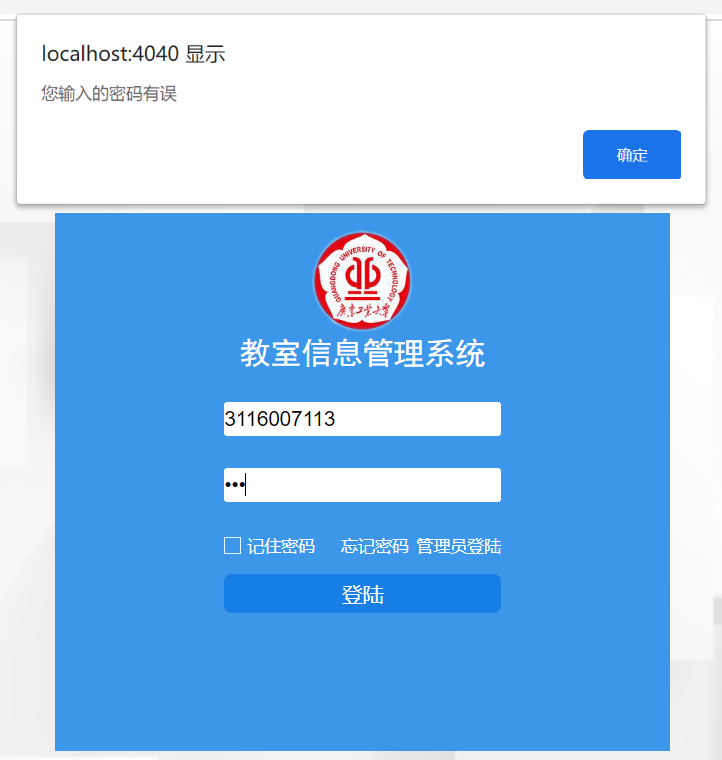


图 3.5 密码错误

1. 学号/教工号输入框的账号存在，且密码匹配成功，进入教室详情页

其主要程序设计如下：

|  |
| --- |
| <template>  <div class="login" @keyup.enter="Login(StudentNum)">  <div class="bgc">  <img :src="imgSrc" width="100%" height="100%" alt />  </div>  <div class="content">  <img :src="topImg" alt />  <p class="title">教室信息管理系统</p>  <br />  <input type="text" placeholder="学号/教工号" class="inputBox" v-model="StudentNum" >  <br />  <input type="password" placeholder="密码" class="inputBox" v-model="StudentPassword" />  <br />  <p class="buttonSel">  <span class="container">  <input type="checkbox" id="remCard" ref="RemInputBox" />  <span class="checkmark"></span>  <label for="remCard" @click="remberPassword()">记住密码</label>  </span>  <span>  <a class="forget" @click="ForgetPassword">忘记密码</a>  <a href="/login/mangerLogin">管理员登陆</a>  </span>  </p>  <button class="loginBtn" @click="Login(StudentNum)">登陆</button>  </div>  </div>  </template>  <script>  export default {  data() {  return {  imgSrc: require("../assets/loginbag.jpg"),  topImg: require("../assets/timg.png"),  LoginMsg: [], //接口接收的登陆用户数据  StudentNum: this.$cookies.get("Account"), //获取cookies,从界面输入的账号  StudentPassword: this.$cookies.get("password"), //从界面输入的密码  PasswordFlag: JSON.parse(this.$cookies.get("PasswordFlag")) //是否要记住密码  };  },  methods: {  // 从后端接口获取登陆用户的信息  getNewsInfo() {  this.$http.get("http://localhost:8080/login/Loginn").then(result => {  if (result.status === 200) {  console.log(result);  this.LoginMsg = result.body;  console.log(this.LoginMsg);  }  });  },  // 登陆按钮  Login(StuNum) {  var StuNumStr = StuNum.toString();  var pasFlag = this.StudentPassword.toString() === this.LoginMsg[StuNum];  if (!(StuNumStr in this.LoginMsg)) {  alert("您输入的学号/教工号不存在");  } else if (this.StudentPassword == "") {  alert("请输入密码");  } else if (!pasFlag) {  alert("您输入的密码有误");  } else {  this.$cookies.set("Account", StuNum); //存cookies  this.$cookies.set("identity","student")  if (this.PasswordFlag) {  this.$cookies.set("PasswordFlag", this.PasswordFlag);  this.$cookies.set("password", this.StudentPassword);  } else if (!this.PasswordFlag) {  this.$cookies.set("password", "");  this.$cookies.set("PasswordFlag", this.PasswordFlag);  }  this.goInfo();  }  },  // 跳转到教室详情页  goInfo() {  this.$router.push("/classroomMsg");  },  // 忘记密码  ForgetPassword() {  alert("请联系老师");  },  // 是否需要记住密码  remberPassword() {  this.PasswordFlag = !this.PasswordFlag;  },  // 根据cookie使记住密码选择框勾选或不勾选  InitRemPasInputBox() {  this.$nextTick(() => {  if (this.PasswordFlag === true) {  this.$refs.RemInputBox.setAttribute("checked",'')  }})    }  },  created() {  if (this.$cookies.get("PasswordFlag") === null) {  this.$cookies.set("PasswordFlag", false);  }  this.InitRemPasInputBox()  this.getNewsInfo()  },  components: {}  };  </script>  <style lang="scss" scoped>  .login {  display: flex;  justify-content: center;  .bgc {  width: 100%;  height: 100%; /\*\*宽高100%是为了图片铺满屏幕 \*/  z-index: -1;  position: absolute;  }  .content {  width: 400px;  height: 350px;  z-index: 1;  position: absolute;  background-color: #3c97ea;  color: white;  display: flex;  flex-direction: column;  align-items: center;  margin-top: 10%;  .title {  font-size: 20px;  font-family: SimHei;  }  img {  width: 70px;  margin-top: 10px;  }  .inputBox {  width: 180px;  height: 20px;  border: none;  border-radius: 2px;  outline: none;  }  .buttonSel {  width: 180px;  font-size: 11px;  display: inline-flex;  justify-content: space-between;  cursor: pointer;  .container {  display: block;  position: relative;  padding-left: 15px;  margin-bottom: 12px;  }  /\* Hide the browser's default checkbox \*/  .container input {  position: absolute;  height: 0;  width: 0;  opacity: 0;  appearance: none; /\*清除默认样式\*/  -webkit-appearance: none;  }  /\* Create a custom checkbox \*/  .checkmark {  position: absolute;  top: 2px;  left: 0;  height: 10px;  width: 10px;  border: 0.5px solid rgb(223, 252, 252);  }  .container input:checked + .checkmark:after {  content: "";  position: absolute;  left: 3px;  top: 0;  width: 2px;  height: 6px;  border: solid white;  border-width: 0 2.5px 2.5px 0;  -webkit-transform: rotate(45deg);  -ms-transform: rotate(45deg);  transform: rotate(45deg);  }  input {  vertical-align: middle;  }  label {  cursor: pointer;  }  a {  text-decoration: none;  margin-left: 5px;  color: white;  }  }  .loginBtn {  outline: none;  border: none;  width: 180px;  height: 25px;  background-color: #177ee5;  color: white;  border-radius: 5px;  cursor: pointer;  }  }  }  </style> |

#### 3.2.1.3 管理员登陆页面的设计

管理员登陆界面主要继承学生/教工登陆页面的设计，区别在于管理员登陆界面只有两个输入框和一个登陆按钮，省去了操作选择框部分，主要考虑管理员身份权限较高，若记住密码可能会被攻击账户信息，所以为了避免一些外来攻击等情况，保证管理员的身份信息不泄露，确保网站的安全，此页面取消了记住密码的复选框。



图 3.6 管理员登陆界面

#### 3.2.1.4 教室详情页的设计

教室详情页主要用来处理各种请求信息，是本次设计的核心，故为了系统的美观性，我采用了element-UI来搭建本页面。

### 3.2.2 后端程序设计

本次后端程序的设计主要承担提供数据接口的任务，使用java语言，开发servlet本地接口。

#### 3.2.2.1 学生/教工信息接口的设计

学生/教工信息接口主要通过接受前端项目发送的get请求，将程序中的学生/教工号的账号和密码以json对象的形式响应给前端。将项目放到Tomcat服务器中运行，并用本项目中的index.jsp测试时，该接口能成功接受请求，而且能够成功响应，其响应成功返回的具体信息如下：



图3.7 项目中index.jsp测试接口的返回数据格式

但是在vue项目中调用接口时，浏览器报了以下错误：

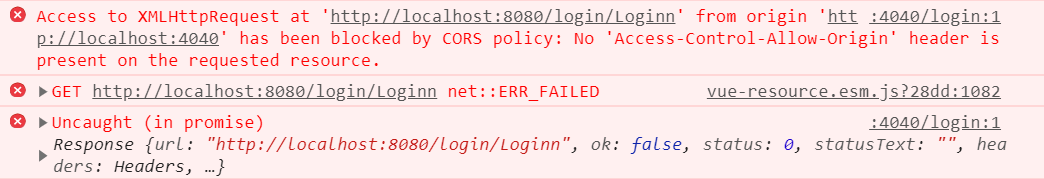


图 3.8 vue项目调用接口时报错

经查信息，发现出现该错误的原因是vue在本地项目中直接请求接口的会被跨域请求拦截。虽然原本的前端项目和后端项目的地址只是端口号不同，但是端口不同已经是属于跨域请求了，解决这个问题的办法之一是在谷歌浏览器的快捷方式的目标属性后边添加以下内容：“ --args --disable-web-security --user-data-dir=D:\temp”。这样相当于用了两个浏览器了。虽然这种方法比较麻烦，每次运行项目都要在浏览器的快捷方式后边加上一句语句，但是实现了本次设计的一大核心部分——后端接口。

解决了跨域问题之后，接口就能在前端项目中调用了，在发送get请求之后就能成功接到以下的响应信息：

|  |
| --- |
| Response {url: "http://localhost:8080/login/Loginn", ok: true, status: 200, statusText: "", headers: Headers, …}  body: {\_\_ob\_\_: Observer}  bodyText: "{"3116007131":"11223344","3116007113":"147852369","3116007128":"123456789ABC","3116007115":"987654321"}"  headers: Headers {map: {…}}  ok: true  status: 200  statusText: ""  url: "http://localhost:8080/login/Loginn"  data: (...)  \_\_proto\_\_: Object |

学生/教师信息接口的各部分主要代码设计如下：

|  |
| --- |
| Web.xml配置文件：  <display-name>login</display-name>  <welcome-file-list>  <welcome-file>index.html</welcome-file>  <welcome-file>index.htm</welcome-file>  <welcome-file>index.jsp</welcome-file>  <welcome-file>default.html</welcome-file>  <welcome-file>default.htm</welcome-file>  <welcome-file>default.jsp</welcome-file>  </welcome-file-list>    <servlet>  <servlet-name>Loginn</servlet-name>  <servlet-class>servlet.Login</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>Loginn</servlet-name>  <url-pattern>/Loginn</url-pattern>  </servlet-mapping>  Servlet项目Login.java：  package servlet;  import java.io.IOException;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import org.json.JSONObject;  /\*\*  \* Servlet implementation class Login  \*/  @WebServlet("/Login")  public class Login extends HttpServlet {  private static final long serialVersionUID = 1L;    /\*\*  \* @see HttpServlet#HttpServlet()  \*/  public Login() {  super();  // TODO Auto-generated constructor stub  }  /\*\*  \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  \*/  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  // 设置请求和响应的编码格式为统一的UTF-8编码格式，避免乱码  request.setCharacterEncoding("UTF-8");  response.setCharacterEncoding("UTF-8");    JSONObject data = new JSONObject();  try {  data.put("3116007128", "123456789ABC");  data.put("3116007115", "987654321");  data.put("3116007113", "147852369");  data.put("3116007131", "11223344");  }  catch (Exception e) {};    response.getWriter().write(data.toString());  response.getWriter().flush();  response.getWriter().close();  }  /\*\*  \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  \*/  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  // TODO Auto-generated method stub  doGet(request, response);  }  }  测试页面index.jsp：  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>  <title>Insert title here</title>  <script>  ajaxGetData = **function** () {  **var** xhr = **new** XMLHttpRequest();  **var** url = "Loginn";  xhr.open("GET",url, **true**);  xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");  xhr.send();  xhr.onreadystatechange = **function**() {  **if**(xhr.status != 200 || xhr.readyState == 4) {  **if**(xhr.readyState == 4 && xhr.status == 200){  **var** data = xhr.responseText;  console.log(data);  **var** json = eval ('(' + data + ')');  document.getElementById("showData").innerHTML += "key=" + "data" + ",value = " + json.data;  }**else** {  alert("出错了：" + xhr.status);  }  }  }    }  </script>  </head>  <body>  <input type=*"button"* value=*"获取借口返回的数据"* onclick="javascript:ajaxGetData()">  <div id=*"showData"*></div>  </body>  </html> |

#### 3.2.2.2 管理员信息接口的设计

管理员信息接口的开发与学生/教师信息接口的开发基本一致，都是用servlet接受get请求并且响应信息，在这里就不赘述了。具体代码如下：

|  |
| --- |
| package administratorLogin;  import java.io.IOException;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import org.json.JSONObject;  /\*\*  \* Servlet implementation class AdministratorLogin  \*/  @WebServlet("/AdministratorLogin")  public class AdministratorLogin extends HttpServlet {  private static final long serialVersionUID = 1L;    /\*\*  \* @see HttpServlet#HttpServlet()  \*/  public AdministratorLogin() {  super();  // TODO Auto-generated constructor stub  }  /\*\*  \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  \*/  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  // TODO Auto-generated method stub  request.setCharacterEncoding("UTF-8");  response.setCharacterEncoding("UTF-8");    JSONObject data = new JSONObject();  try {  data.put("GDUT01", "123456789");  data.put("GDUT02", "987654321");  }  catch (Exception e) {};    response.getWriter().write(data.toString());  response.getWriter().flush();  response.getWriter().close();  }  /\*\*  \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  \*/  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  // TODO Auto-generated method stub  doGet(request, response);  }  } |

经过测试，以上代码能够正确编译并运行在服务器上，保证了前后端数据正常的连接。