江 西 理 工 大 学

本 科 毕 业 设 计（论文）

题　　目：基于微信小程序的图书馆座位管理系统

专题题目：

学　　院：信息工程学院

专　　业：计算机科学与技术

班　　级：16 计算机 3 班

学　　号：1520163312

学　　生：李风杰

指导教师：曾鹏程　职称：实验师

时间：　2020 年 5 月 20 日

摘　要

随着各大高校的不断扩招与考研人数的逐年上升，图书馆座位资源日趋紧张。高校图书馆内的座位资源数量相对固定，如何高效管理与使用图书馆内有限的座位资源成为高校图书馆管理的重要课题。在这种背景下，本文提出基于微信小程序的图书馆座位管理系统，为图书馆座位管理提供高效的解决方案。

本文全面分析了图书馆座位管理系统的需求，进行了系统的设计与实现。系统手机客户端使用微信小程序呈现，管理员管理端采用 Vue 和 ElementUI 框架实现。后端采用 Spring Boot、MyBatis 技术和 Redis 缓存技术，使用了 MySQL 数据库存储系统数据。系统主要模块有阅览室管理模块、座位管理模块、扫码入座模块。学生可以通过微信小程序查看图书馆内实时的座位使用情况、扫码落座。图书馆管理员可根据汇总后的数据进行座位的管理和使用工作。

基于以上技术和设计方案的实现，本文完成了基于微信小程序的图书馆座位管理系统的设计与开发。最后对系统进行了功能和页面测试，系统各功能模块均通过测试。

关键词：微信小程序；图书馆；座位管理

ABSTRACT

With the continuous expansion of enrollment in major universities and the increasing number of graduate students, the library seating resources are becoming increasingly tense. The number of seat resources in the university library is relatively fixed. How to efficiently manage and use the limited seat resources in the library has become an important issue in university library management. Under this background, the library seat management system based on WeChat applet is proposed here to provide an efficient solution for library seat management.

This article comprehensively analyzes the needs of the library seat management system, and designs and implements the system. The mobile phone client of the system is presented by WeChat applet, and the administrator management end is implemented by Vue and ElementUI framework. Using Spring Boot, MyBatis technology and Redis cache technology, the MySQL database is used to store system data.The main modules of this system include a reading room management module, a seat management module, and a code scanning module. Students can check the real-time seat usage inside the library through WeChat applet, scan the code and sit down. Librarians can manage and use seats based on the aggregated data.

Based on the implementation of the above technology and design scheme, the design and development of the library seat management system based on WeChat applet is completed here. Finally, the system was tested for functions and pages, and each functional module of the system passed the test.

Keywords: WeChat Mini Programs; library; Seats management

目 录

第一章　绪论

1.1 选题背景与研究目标

1.1.1 选题背景

根据 2019 中国统计年鉴数据显示，2018 年中国普通本专科在校人数达到2831 万，相比 2017 年中国普通本专科在校人数 2753 万有所上升[1]。随着各大高校的不断扩招，各大高校图书馆座位资源相继出现座位资源使用紧张的情况。虽然各大高校针对这种情况做出增加座位硬件资源的举措，但是总体上，增加的座位资源有限，并没有改变各大高校图书馆座位资源使用紧张的局面。

近年来考研人数不断攀升，2018 年 12 月 23 日，这一年的考研人数，再一次变成了“历史新高”。从教育部公布的数据来看，2019 年全国考研报名人数达到了史无前例的290万人，比2018年考研人数增加了52万人，增幅达到了21.8%，考研增加人数和增长率均为近年来最高。近 5 年，考研人数逐年增长，同 2015 年的 165 万人相比增加了 125 万人[2]。图书馆自习室无疑是考研学生复习备考的最佳场所，考研人数的逐年攀升使得各大高校图书馆座位硬件资源更加紧张。另外大学生考证考级热度的不断“升温”，在一定程度上使得高校图书馆座位资源更加紧张。

在高校图书馆座位硬件资源数量有限的情况下，高校图书管中出现不同程度的占座现象，在图书馆常常出现有物无人的座位。这使得需要使用座位的同学无法使用这些有物无人的座位资源，这些座位处于闲置状态。占座现象过多时，造成图书馆整体可用的座位资源处于一种“假满”状态。这种占座行为浪费了图书馆有限的宝贵的座位资源，大大降低了图书馆座位资源的使用效率。

《2019 年微信数据报告》显示，微信月活跃账户数超 11.5 亿，相比 2018 年增长 6%，微信小程序用户数量达到 3 亿。可见，微信在网民中普及程度十分高，微信小程序也不断为人们所接受。2017 年 1 月微信正式发布了微信小程序，不需要下载、不需要安装、用完即走、无须卸载，且开发微信小程序由微信平台审核，安全性有保障，还可以跨平台使用[3]，不管是 Android 系统还是 iOS 系统，使用同一个版本即可[4]，这些特点为高校图书馆通过微信小程序技术方式为高校学生、图书馆管理员提供座位管理服务奠定了技术基础。

综合以上背景，本文提出基于微信小程序的图书馆座位管理系统。解决高校图书馆座位有书无人，学生无法落座、学生临时离开，书本可能被视为占座清除、学生一座难求，馆方压力倍增的难题。在高校图书馆座位资源相对固定的情况下，

提升高校图书馆座位资源的使用效率，降低高校图书馆座位需求压力。高效的高校图书馆座位管理方案。

1.1.2 研究目标

本课题研究目标主要有以下几个方面：

1) 降低高校图书馆座位人力、财力及管理成本
2) 解决高校图书馆座位有书无人，学生无法落座现象
3) 解决学生临时离开，书本可能被视为占座清除的问题
4) 解决学生一座难求，馆方压力倍增的难题
5) 降低高校图书馆座位资源不足与同学对座位资源需求日益上升的矛盾
6) 使得管理图书馆座位资源像管理图书馆图书一样高效实用
7) 提升高校图书馆良好的阅读学习氛围

1.2 课题研究意义

对于高校图书馆来说，本课题提高了高校图书馆中相对有限的座位硬件资源的使用效率，改变了图书馆座位资源“假满”的现象。充分利用每一个座位硬件资源的价值，做到了“零占座”。只要图书馆还有无人正在使用的座位，图书馆就可以继续容纳学生，所有座位都充分得到了使用。在高校图书馆座位硬件资源未增加的情况下，提升了高校图书馆可服务学生的数量，使得高校图书馆在一定时间内容纳空间上得到“提升”。图书馆每一间自习室和阅览室座位使用情况都可以通过本系统一目了然的展示出来，与传统图书馆管理员人工巡视相比，降低了图书馆管理员进行巡视管理的时间成本。

学生临时离开，书本很可能被图书馆管理员视为占座而清除，这种情况会给同学带来不必要的麻烦。通过本课题的研究，系统彻底解决了此类问题，同学再也不用担心因短暂的离开自己的书本被图书馆管理员清除的问题。另外因为管理员可以通过此系统一键查看图书馆座位使用情况，减少了巡视管理次数，降低了因巡视而对学生学习的干扰影响，学生在图书馆学习的专注程度有所提升。

在整个学校范围来看，通过本课题的研究，系统高效的图书馆座位管理降低了学习对图书馆硬件资源的资金投入，降低了维护成本。学生养成一人一座、离开退座、专注学习的良好习惯。有益于学生综合素质的培养。

本课题将微信小程序技术应用到高校图书馆座位管理，当下热门应用技术与校园实际问题相结合，采用面向对象的实现方法，设计实现了高效实用的高校图书馆座位管理系统，具有一定的学术价值。

1.3 国内外研究现状

根据查阅文献与调查发现，当前国内外高校图书馆自习室管理系统主要存在以下几种：

1） 基于校园卡图书馆管理系统

在高校图书馆座位管理方式中，由校园学生卡为基础的管理方式是使用较早，使用范围较为广泛的一种方式。这种方式主要依托于学生的“校园一卡通”，学生通过刷校园卡通过高校图书馆的闸机。该方式对学生身份进行验证，验证是该校学生或者为该图书馆的注册会员即可进入图书馆。这种方式只是简单的对入馆人员进行了身份验证，限制了无关人员进馆，在一定程度上降低了图书馆的访问流量。但是，总的来说，基于传统校园卡的图书馆管理系统并非真正意义上的图书馆座位管理系统，它无法做到管理图书馆中每一间自习室和阅览室的每一张座位。

2） 基于硬件的图书馆座位管理系统

关于基于硬件的图书馆座位管理系统，有学者也不断研究此类问题，比如有基于 STC90C52 单片机的图书馆座位管理系统[5]、基于 GIS 技术的高校图书馆座位管理系统[6]、基于单片机及 CAN 技术的图书馆自习室座位管理系统的实现[7]、基于压力传感的图书馆座位管理系统[8]等，基于硬件的图书馆座位管理系统利用各种硬件传感器对高校图书馆座位的人员使用情况进行采集，对使用座位的学生进行身份验证。这种座位管理方式，采用硬件采集数据，这无疑给高校图书馆增加了资金成本。另外，硬件数据展示的局限性也给同学使用图书馆座位资源带来很多麻烦。

3） C/S 结构（客户端/服务器结构）图书馆座位管理系统

在这里客户端/服务器结构中客户端指的是手机客户端，比如 Android 客户端。有学者研究基于 Android 平台的图书馆座位管理系统界面设计与实现[9]，使用 Android 客户端需要用户下载安装包安装后才可以使用，这对于学生用户并不友好，甚至可以说因为客户端的“笨重”限制了其在高校图书馆座位管理方面的研究与发展。

4） B/S 结构（浏览器/服务器结构）图书馆座位管理系统

这种 B/S 结构主要分为两个方面，第一，基于 JSP、HTML 等网页技术的 B/S 结构，比如有基于 SSH 框架的图书馆自习室管理系统的设计与实现[10]，以及采用 HTML 等网页技术的图书馆移动座位管理信息系统设计与实现[11]；第二，基于微信公众号的 B/S 结构，与一般的浏览器/服务器结构相比，基于微信公众号的 B/S 结构依托于微信平台，启用微信平台的开发者需要申请服务器资源，用于存放自己开发的程序文件[12]。

5） 基于微信小程序的座位管理系统

也有学者在基于微信小程序的图书馆座位管理系统的方向上研究[13]，如基于微信小程序的图书馆座位预约系统的设计与实现[4]等课题。

1.4 研究内容及组织结构

1.4.1 研究内容

本文根据各大高校图书馆所遇到的座位管理方面的难题，设计并实现了一款基于微信小程序的图书馆座位管理系统。系统一方面解决了困扰图书馆方“占座”现象，减少了图书馆管理员的巡视次数，降低了馆方的运营维护成本[14]。另一方面系统方便了学生寻找图书馆空位，大大降低了学生在馆内寻找空位的时间。降低了学生因寻找座位而产生的焦虑情绪，一定程度上提升了学生图书馆内的读书学习效率。

1.4.2 组织结构

本文的结构主要如下：

第一章 绪论 主要交代了本文的选题背景、研究目标，课题研究意义，课题国内外研究现状。以及研究内容与论文的组织结构。

第二章 基础理论和相关技术介绍 介绍了系统使用的相关技术及其基础理论。

第三章 系统需求分析 这一章对系统自定义排座、座位状态查询、扫码入座等具体功能进行了详细的分析。

第四章 系统设计 这一章的内容有系统的设计原则、总体设计以及用户模块设计、阅览室管理模块设计、座位管理模块设计等各功能模块的设计。

第五章 系统实现及应用效果 这一章主要展示了系统最终运行使用效果。

第六章 总结与展望 对整个系统进行总结和展望。

第二章　基础理论和相关技术介绍

2.1 Spring Boot技术

Spring Boot技术是由Pivotal团队开发的在Java EE编程语言一个轻量级开发框架。它基于Spring框架的4.0版本开发设计，能快速地创建独立的、可生产发布的且能够直接运行的应用程序[15]。通过Spring Boot，可以轻松地创建独立的，基于生产级别的基于Spring的应用程序，我们可以“运行”这些Spring Boot应用程序。Spring Boot应用程序一般不需要太多的Spring配置。独立的Spring应用程序可以直接嵌入Tomcat。

2.2 MyBatis技术

MyBatis是一个基于Java的数据持久层(ORM)框架，支持自定义SQL，存储过程和高级映射。MyBatis消除了几乎所有的JDBC代码以及参数的手动设置和结果检索[16]。MyBatis可以使用简单的XML或注释进行配置，并将图元、映射接口和Java POJO（普通的旧Java对象）映射到数据库记录[17]。MyBatis几乎可以代替JDBC，是一个支持普通SQL查询，存储过程和高级映射的基于Java的优秀持久层框架[18]。

2.3 Redis技术

Redis 是一个开源的使用ANSI C语言编写、遵守BSD协议、支持网络、可基于内存亦可持久化的日志型、Key-Value 数据库，并提供多种计算机开发语言的API的非关系型数据库。Redis通常被称为数据结构服务器。这意味着Redis通过一组命令提供对可变数据结构的访问，这些命令是使用具有TCP套接字和简单协议的 C/S(服务器-客户端)模型发送的。因此，不同的进程可以以共享的方式查询和修改相同的数据结构。在 Redis 中实现的数据结构具有一些特殊的属性：Redis始终将它们存储在磁盘上，即使它们总是被服务并修改到服务器内存中也是如此。这意味着Redis速度很快，但这也是非易失性的。数据结构的实现会提高内存效率，因此与使用高级编程语言建模的相同数据结构相比，Redis内部的数据结构可能会使用较少的内存。Redis提供了许多自然可以在数据库中找到的功能，例如复制，持久性的可调级别，集群，高可用性。其操作不仅是SET和GET，而且还可以使用复杂数据类型（如列表，集合，有序数据结构等）进行操作。

2.4 Vue 框架

Vue.js 是一套数据驱动的用于构建用户界面的，可以自底向上逐层应用的渐进式前端框架。相比于其他主流的 Java Script 框架例如 Angular JS 或 React 等，Vue.js 具有运行效率高、语法简洁、自身占用空间小、上手容易等特点[19]。Vue.js 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。

2.5 微信小程序架构

微信小程序是从微信中的 WebView 和 JS-SDK 发展演化到了今天的形态。微信小程序和普通的 H5 网页主要有以下几个方面的区别：

1） 运行环境

小程序基于浏览器内核重构的内置解析器，而 H5 的宿主环境是浏览器。小程序中没有 DOM 和 BOM 的相关 API。

2） 系统权限

小程序能获得更多的系统权限，如网络通信状态、数据缓存能力等[20]。

3） 渲染机制

小程序的逻辑层和渲染层是分开的，而 H5 页面 UI 渲染跟 JavaScript 的脚本执行都在一个单线程中，互斥。所以 H5 页面中长时间的脚本运行可能会导致页面失去响应[21]。微信小程序架构图如图 2-1 所示。

2.6 MySQL 数据库

MySQL 是最流行的关系型数据库管理系统，在 WEB 应用方面 MySQL 是最好的 RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。MySQL 的创始人以 MySQL 为基础，成立分支计划 MariaDB。而原先一些使用 MySQL 的开源软件逐渐转向 MariaDB 或其它的数据库。支持 Windows、AIX、BSDI、NetBSD、FreeBSD、Linux、Mac OS、OS/2 Wrap、Novell、NetWare、HP-UX、OpenBSD、Solaris 等多种操作系统。MySQL 为多种编程语言提供了相应的 API 工具。这些编程语言包括 C、C++、PHP、Python、C#、Delphi、VB.NET、Perl、Ruby、Eiffel 和 Java 等。MySQL 支持多线程，可以充分利用 CPU 资源，支持多用户。MySQL 优化的 SQL 查询算法，有效地提高查询速度。既能够作为一个单独的应用程序在客户端服务器网络环境中运行，也能够作为一个程序库而嵌入到其他的软件中。常见的编码如中文的 GB2312、日文的 Shift JIS、BIG5 等都可以用作数据表名和数据列名。提供 JDBC、ODBC、和 TCP/IP 等多种数据库连接途径。可以处理拥有上千万条记录的大型数据库[16]。JAVA 程序通过 JDBC 方式与 MySQL

进行连接。

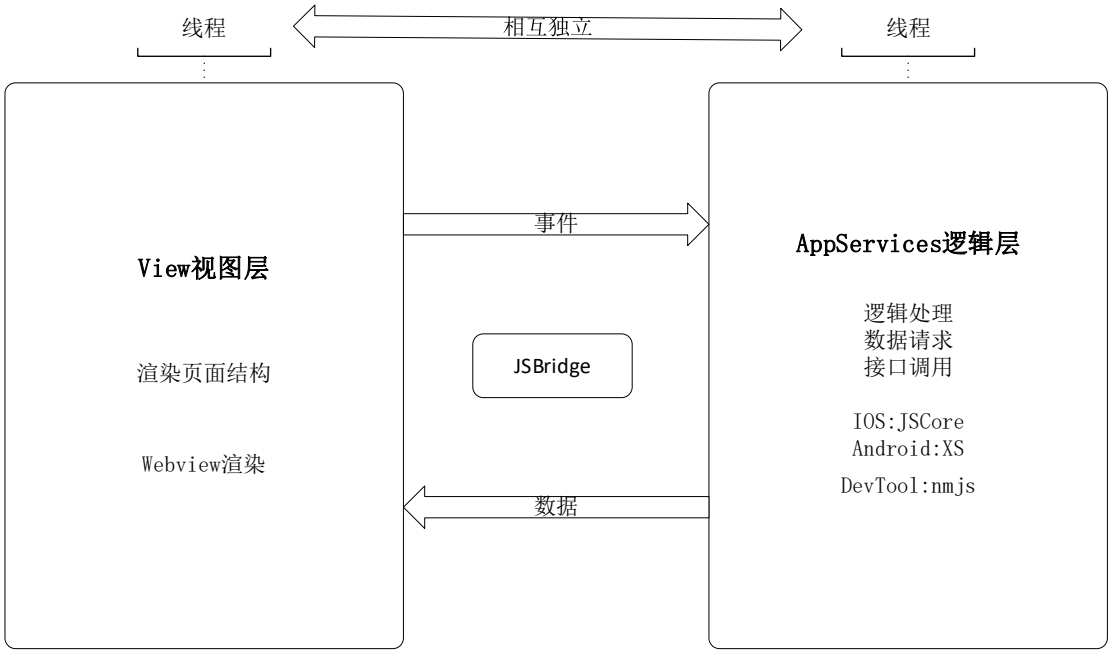


图 2-1 微信小程序架构图

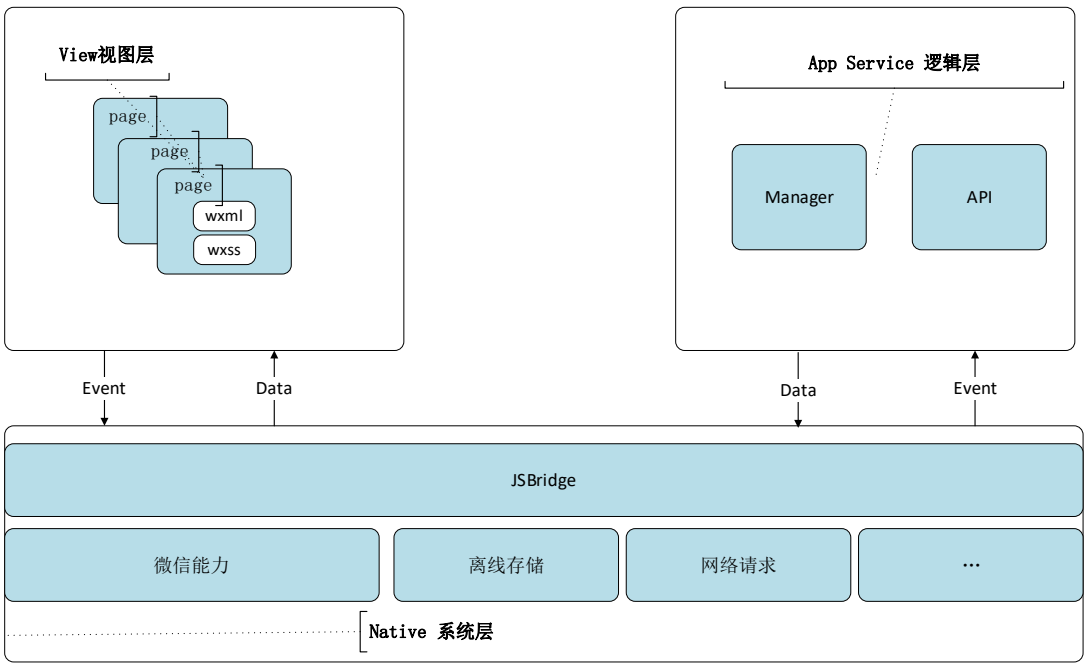


图 2-2 微信小程序架构层次图

在微信小程序开发过程中我们面向的是 iOS、Android 微信客户端和 PC 版本微信，这体现了微信小程序介于 Web 端和原生 App 之间，能够丰富调用功能接

口，同时又具有真正意义上的跨平台性。微信小程序架构层次图如图 2-2 所示。小程序是有自己的组件的，这些基本组件就是基于 Exparser 框架。Exparser 基于 WebComponents 的 ShadowDOM 模型，但是不依赖浏览器的原生支持，而且可在纯 JS 环境中运行。小程序中，所有节点树相关的操作都依赖于 Exparser，包括 WXML 到页面最终节点树的构建、CreateSelectorQuery 调用和自定义组件特性等。在内置组件中，有一些组件并不完全在 Exparser 的渲染体系下，而是由客户端原生参与组件的渲染。比如说 Map 组件。它渲染的层级比在 WebView 层渲染的普通组件要高。引入原生组件的优点有：扩展 Web 的能力、体验更好，减轻 WebView 的渲染工作、绕过 SetData、数据通信和重渲染流程，性能更好。

2.7 本章小结

本章主要介绍了系统涉及的 Spring Boot、MyBatis、Redis 等相关技术和相关理论。详细介绍了每一种技术的研究及发展现状、技术的优点以及缺点等。本章对每一种技术分别展开进行了详细介绍。

第三章　系统分析

3.1 基本情况

目前各大高校图书馆对座位管理普遍使用传统的管理员巡视的方式，图书馆内座位得不到较为科学的管理，无法使图书馆容纳更多的需要座位学习的学生。管理员无法高效便捷的获取图书馆实时的座位使用情况，面对“占座”行为，只能暂时对有物品而无人使用的座位上的物品进行临时的收纳，这不仅给图书馆管理员带来巨大的工作压力，而且也会对因打热水等不可抗拒原因短暂离开座位的同学物品造成误缴。对物品的收纳也会给高校图书馆方增添物品的管理看护工作，增加因图书馆座位资源不足而造成的学生与图书馆方管理的矛盾。长期而言这种矛盾会影响学生和图书馆管理员双方的工作与学习效率。因此如何解决管理员与学生无法即时获取到图书馆内座位使用情况成为系统需要讨论研究的重要问题。另外如何在高校内图书馆座位硬件资源有限的情况下，提高每个座位的使用效率，以达到服务更多需要在馆内座位学习的学生的问题也是本系统课题的重要需求。

3.2 座位管理系统需求分析

近些年来的“考研热”、“考证热”现象使得高校图书馆人流量剧增，各大高校图书馆座位硬件资源紧张。高校内出现学生早晨在图书馆开馆前排起长龙，开馆后迅速将图书馆座位一抢而空的现象。在这看似“一抢而空”的背后存在着不少“假空”现象，即“占座”行为，图书馆座位处于有物无人的状态，这些座位在一定的时间内处于闲置而不能使用的状态，浪费了其原本的价值，使得图书馆的座位资源更加紧张。

因各大高校学生对图书馆座位需求的提升，造成高校图书馆座位资源供给不足[22]。各大高校图书馆面临在高校图书馆内的座位硬件资源一定的情况下，如何提高每一座位的使用效率，防止图书馆内的“占座”行为，成为座位管理系统的重要需求。

3.3 座位管理系统功能需求分析

3.3.1 自定义排座功能

全国各大高校图书馆的每一间阅览室和自习室的座位摆放位置都不尽相同，而且在不同的时间段，阅览室或者自习室的座位布局摆放可能发生变化。于是，

针对这种情况，系统需要提出便捷实用的解决方案。只用如此，系统才可以提供图书馆内座位实时的状态信息，用于向图书馆管理员以及需要入馆学习的学生服务。所以说自定义、灵活、可操作、可修改的排座功能为整个图书馆座位管理系统的基础与保证。

3.3.2 座位状态查询功能

座位状态查询功能要求能够即时查询每一间阅览室的座位使用情况，每一个座位的当前状态[23]。系统最终可视化的将所有阅览室座位使用情况展示给图书馆管理员和学生用户。图书馆管理员可以随意切换查看本学校图书馆阅览室和自习室座位使用情况。学生可以根据状态使用图书馆中的空座，图书馆管理员可以查看正在使用的座位学生个人信息，座位信息进行管理工作。使得管理员可以高效的管理图书馆的座位资源，学生可以快速的找到合适的座位入座。

3.3.3 座位状态更新功能

因为系统需要能够实时展示图书馆中每一间阅览室中座位的状态，所以这就要求系统实现每一个座位状态的更新功能。主要有三方面的状态转换方式：

1） 学生使用座位发生的座位状态更新

学生找到并选择扫描图书馆空位置座位的二维码后，选择学习或离开等状态，座位状态需要改变。学生在对有物无人的座位进行举报时，该座位的状态也需要发生改变，以指引管理员对图书馆内占座行为进行管理。

2） 管理员管理引起更新的座位状态

图书馆管理员一方面可以对出现故障等问题的座位进行修改状态，另一方面管理员会根据同学上报的有物无人的座位上物品进行处理后更新座位的状态。

3） 时间超时状态自动更新

学生扫码选择使用座位需要选择使用的时间长度，比如 30 分钟、60 分钟等。时间过后座位的状态将从使用状态转换为空座状态，座位的所属者也将从扫码的学生转变为无人使用的状态。

3.3.4 扫码入座功能

扫码入座功能是本系统管理座位的核心手段，学生扫描二维码将图书馆的座位与互联网技术系统后台连接，将图书馆的座位数据化，最终以可视化的形式展示出来。扫码前需要对学生用户身份以及位置信息进行验证，以确保扫码时该学生位于图书馆内，防止远程扫码行为的发生，保证数据的准确性。

3.3.5 通知与公告功能

在以往传统的方式中，图书馆发布临时闭馆维修、节日图书馆阅览室和自习室安排公告通知一般采用纸质的公告，并张贴在图书馆入口处。这种行为不仅传达效果不好，而且较多人聚集张贴纸质公告处会造成人员拥堵，存在较大的安全隐患。本系统为各大高校提供通知和公告的发布与浏览功能。图书馆管理员可以快速的传达图书馆的通知，学生也可以通过系统方便的了解图书馆的自习室和阅览室开发时间等图书馆的工作安排。

3.3.6 学习历史功能

针对学生的扫码学习历史，本系统提供学习历史功能。学生可以查询自己以往的学习记录。学习记录需要包括学习时间、开始时间、结束时间、学习形式等必要的信息。

3.4 座位系统用户权限需求分析

基于微信小程序的图书馆座位管理系统主要用户角色有图书馆管理员和高校需要使用座位资源的学生用户。针对这两种用户需要有一定的权限划分。

图书馆管理员具体所需的权限有：

1) 图书馆管理员有创建阅览室和自习室并且自定义座位摆放及其状态信息的权限。
2) 图书馆管理员有修改、删除阅览室和自习室并且自定义座位摆放及其状态信息的权限。
3) 图书馆管理员具有查看每一个正在使用座位的学生用户的个人信息的权限。
4) 查看阅览室和自习室中座位状态的权限。
5) 图书馆管理员具有修改每一个座位的使用状态的权限。
6) 发布通知和公告的权限。

学生图书馆座位用户具体所需的权限有：

1) 查看图书馆阅览室和自习室座位状态的权限。
2) 查看个人学习历史记录的权限。
3) 上报有物无人“占座”行为的权限。
4) 扫描二维码选择并使用一定时间座位的权限。
5) 查看图书馆通知和公告的权限。

3.5 本章小结

本章首先对系统基本情况需求进行分析，然后对座位管理需求进行分析、系统每个模块功能需求分析，这里包括自定义排座、座位查询、座位状态更新、扫码入座、通知公告、学习历史功能需求分析。最后对座位系统的用户权限需求进行了具体分析。

第四章　系统设计

4.1 系统设计原则

基于微信小程序的图书馆座位管理系统设计遵循以下原则：

1） 可靠性原则

图书馆座位管理系统的可靠性是系统能够顺利的运行下去的前提和保证，座位状态信息需要是可靠的，只有这样学生用户在使用图书馆座位资源时才有效，数据才能准确的表达图书馆座位使用情况。图书馆座位管理才有意义可言，否则，图书馆座位管理系统存在将毫无意义。

2） 经济性原则

基于微信小程序的图书馆座位管理系统是采用微信小程序和浏览器的软件应用，高校图书馆无需增加任何硬件设备。图书馆管理员和学生用户都可以通过微信来管理座位信息或使用座位。系统遵循经济性原则才能让系统付诸实践，在各大高校中推广开来，实现更大的价值。

3） 易用性原则

基于微信小程序的图书馆座位管理系统的设计遵循易用性原则，无论是管理员用户自定义阅览室座位布局，还是学生用户扫码入座。都可以通过 PC 网页或者微信小程序点击完成。无需进行专业化的训练。所有用户皆可在无新手指引的情况下熟练使用本系统。

4） 可扩展性原则

本系统面向全国各大高校，可以同时服务多个高校多个校区图书馆的座位管理工作。学校数量可以灵活增添。另外，系统开发采用面向对象的软件开发方法，亦具有可扩展性。

5） 业务完整性原则

基于微信小程序的图书馆座位管理系统的设计要考虑到业务完整性，针对各大高校不同布局的阅览室要做完整考虑，对座位的不同状态要做充分考虑。对于高校图书馆通知与公告等其他相关功能也要考虑到。遵循业务完整性原则，考虑不同高校图书馆不同需求，以及每一个业务中不同情况考虑完整。如此，系统运行过程中给图书馆管理员和学生用户才会使用流畅、有良好的用户体验。高校方可以系统的管理图书馆座位及相关工作。

4.2 系统总体设计

本系统主要分为用户管理模块、阅览室管理模块、座位管理模块、扫码入座

模块、学习历史模块和通知与公告模块等六个大的部分：

1） 用户管理模块

这里的用户主要分为图书馆管理员和学生用户，针对这两类不同的用户类型系统将有不同的功能。对于图书馆管理员主要有管理员个人信息的绑定，PC 网页管理端的登录权限等的划分。学生用户个人信息的绑定，用于使用座位时个人身份验证、学习历史的存储等。

2） 阅览室管理模块

高校图书馆管理员可以根据自己学校的阅览室数量进行创建阅览室，自定义阅览室的信息，包括阅览室房间号、阅览室楼层、创建时间、阅览室状态等信息。图书馆管理员可以对阅览室进行添加、修改、删除操作。

3） 座位管理模块

在这个模块中，图书馆管理员可以在 PC 管理网页端自定义阅览室的座位布局以及每一个座位的状态。管理员可以对座位的状态信息、位置信息进行修改操作。在微信小程序端图书馆管理员可以查看每一间阅览室和自习室的座位使用状态、当前使用该座位的学生用户的个人信息。图书馆管理员可以对座位的状态信息进行单独修改。学生用户可以通过微信小程序查看图书馆内每一间阅览室和自习室的座位当前状态，扫描座位二维码验证后使用座位，座位状态发生改变。学生对于图书馆内有物无人的座位可以举报。上报后该座位的状态信息同样发生改变。

4） 扫码入座模块

这个模块主要是让学生扫描二维码后再使用图书馆内的座位资源，在扫码入座前要对学生的身份、位置等信息进行验证。确保在学生扫描使用座位时，学生处于图书馆内，而且当前未扫描使用其他座位；被扫描的座位未被其他人使用并处于可用的状态。从而做到，一个学生用户在同一时间仅可以使用一个座图书馆位，一个图书馆的座位只能被一个人使用。

5） 学习历史模块

学习历史模块用来记录学生用户每次扫码使用座位后的信息。学生可以查阅自己扫码及学习记录。

6） 通知与公告模块

图书馆管理员可以发布通知和公告，学生用户可以在微信小程序中查看管理员发布的公告详情信息。管理员可以在这个模块创建并发布通知公告。发布的通知公告验证通过后将展示在微信小程序前端通知与公告列表中，点击列表可以查看通知与公告详情。

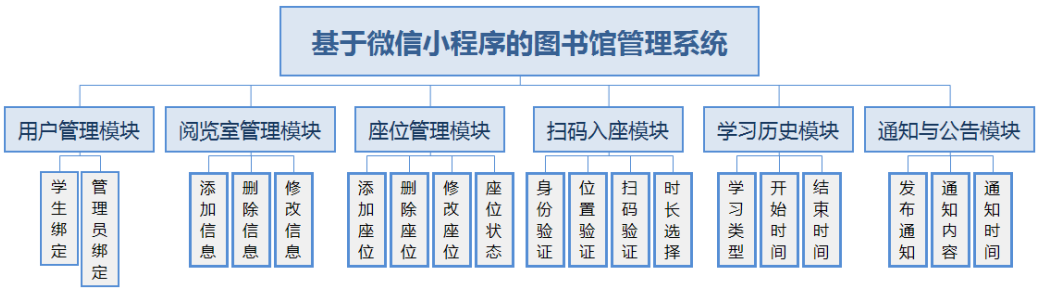


图 4-1　系统功能模块结构图

4.3 各功能模块详细设计

4.3.1 用户管理模块

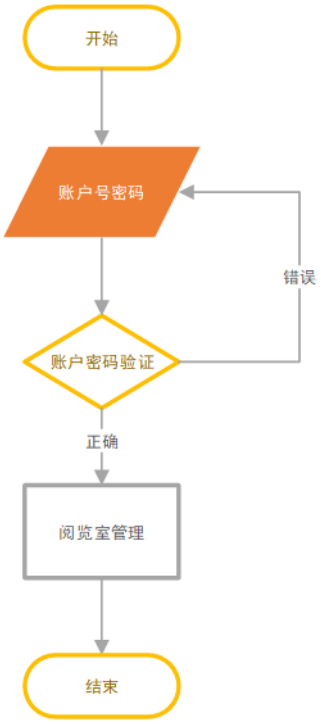


图 4-2　管理员后台登录流程图

这个模块主要是处理图书馆管理员和学生用户登录与信息绑定工作，管理员绑定有管理员姓名、管理员头像、管理员昵称等信息。学生用户绑定学号、姓名、性别、学校、班级、学院、专业、年级。这些管理员或者学生用户的详细信息是通过后台统一导入系统中的，无需在微信小程序端注册。用户管理模块要做的就是把图书馆管理员的微信信息与系统后台导入的个人信息绑定，学生用户的微信信息与管理系统后台导入的学生个人信息向绑定。只有当图书馆管理员绑定微信三方登录信息后，图书馆管理员才可以进行座位和学生的后序管理工作。学生用

户绑定微信信息登录后，才可以查看图书馆座位使用情况，进行扫码使用座位操作。用户管理模块的登录信息绑定保证了各大高校图书馆座位信息的安全性，为图书馆内座位信息状态的实时更新提供了数据保证[24]。

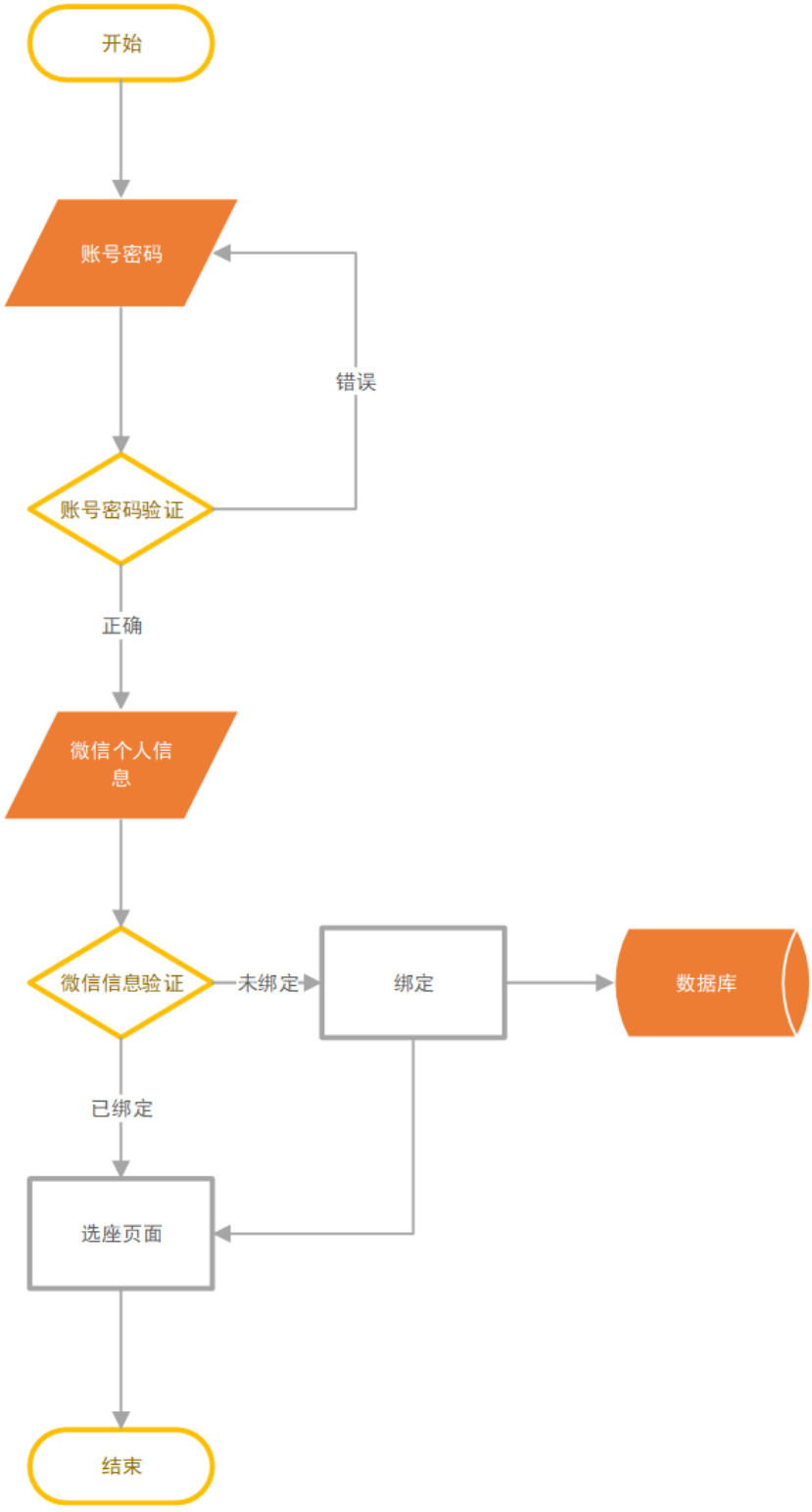


图 4-3　学生用户登录信息绑定流程图

4.3.2 阅览室管理模块

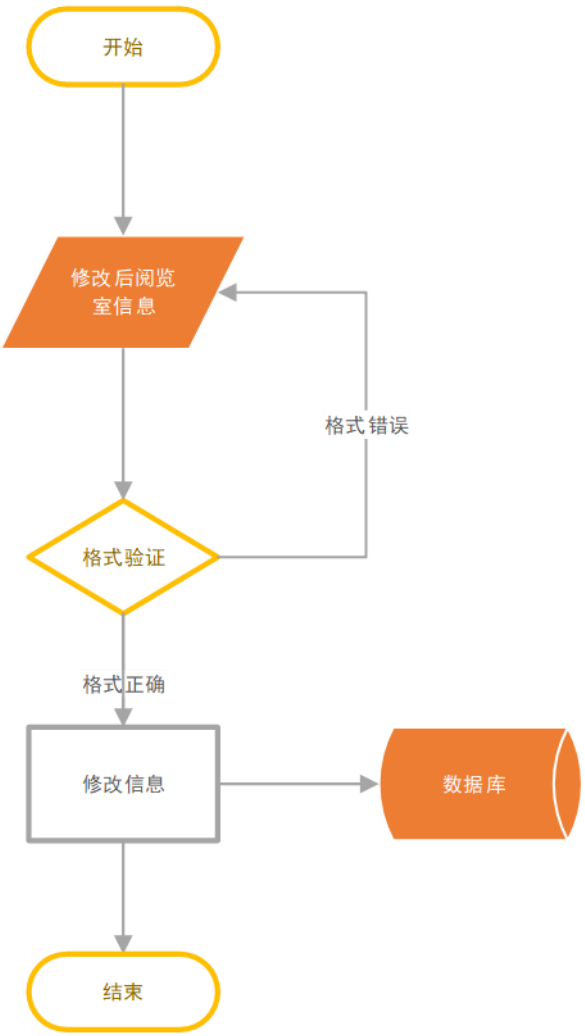


图 4-4　阅览室信息修改流程图

图书馆管理员在电脑网页端以管理员身份登录后，可以通过阅览室（自习室）管理模块对本图书馆的阅览室进行增添、删除、修改操作。阅览室信息包括阅览室所在的楼层号、阅览室或自习室的房间号码、创建日期、上次修改信息日期、阅览室当前状态（正常使用、维修中）等。管理员可以单一删除阅览室或批量删除阅览室。在阅览室信息的修改、增添、删除操作中，操作的数据都需要进行格式验证，保证操作数据的准确性。如图 4-4 所示。这里以修改阅览室信息为例，其他阅览室操作与修改操作类似，这里不再赘述。

4.3.3 座位管理模块

座位管理模块涉及到各大高校图书馆的每一间阅览室和自习室中座位信息的管理。首先，图书馆管理员可以根据自身图书馆阅览室中座位的摆放布局、座

位的大小自定义创建座位。管理员可以创建普通座位、正在维修中座位、过道（如图 4-5 所示）等多种类型的阅览室座位布局。管理员通过电脑浏览器打开 Web 网页可视化点击创建，适应不同阅览室布局、不同高校图书馆的座位种类、座位的不同使用状态。

系统需要维护每一个座位的实时的使用状态，每一个座位的状态随时可能发生改变，每一个座位当前使用它的学生也会发生改变。座位状态主要存在以下几种切换形式：

图 4-5　座位状态图

1）　空位状态转换为有人使用状态

需要使用座位的学生在扫码选择使用座位后，座位会由空位状态转换为有人使用的状态。具体详细介绍请看 4.3.4 扫码入座模块。

2）　空座状态转换为短暂离开状态

学生在需要短暂离开时，可以扫码后选择短暂离开时长，在短暂离开的时间内，座位的使用权不会转移，座位会一直处于离开状态直到超过离开时间，系统会自动将离开转变为空座状态。

3）　空座状态转换为维修状态

当图书馆在发现图书馆座位需要维修时，可以将座位的空位状态设置为维修状态，提醒学生此座位正在维修，防止使用造成安全问题。

4）　空座状态转换为有无无人状态

为了防止图书馆内一些同学的“占座”行为，当其他同学根据小程序座位状态图示寻找到的座位有物品，而且系统显示为空座位状态时，可以认定此座位处于被占座的状态，同学可以在小程序上点击“有物无人 占座举报”按钮一键上报。系统会将座位的空座状态转化为有物无人状态。图书馆管理员可以根据系统更新后的座位状态图，快速准确的寻找到违规被占用的图书馆座位。降低了图书馆的“占座”现象的发生[25]。

5）　有人使用状态转换为空座状态

在学生用户入座时需要选择使用座位的时间，如：三十分钟、六十分钟、一百二十分钟等。超过这个时间段后，系统会自动将座位的有人使用状态转换为空座状态。

6）　短暂离开状态转换为空座状态

与“有人使用状态转换为空座状态”类似，超过选择离开的时间段后，系统会自动将座位的短暂离开状态转换为空座状态。

4.3.4 扫码入座模块

学生通过小程序展示的图书馆座位使用情况状态图示，寻找到可以使用的座位，系统先进行学生身份的验证，验证扫码的用户是否属于本校的学生或者是否是具有本图书馆学习阅览权限的用户。验证通过后系统再对扫码的用户的进行位置验证，系统使用微信小程序提供的位置验证接口获取将要使用座位用户的位置信息，与后台预留的图书馆坐标进行距离测算。如果测算距离小于被允许的范围，系统将允许小程序调起扫码组件，继续扫码验证当前需要入座的座位信息。验证通过后，学生可以选择“学习 30 分钟”、“学习 60 分钟”、“学习 120 分钟”等，学生获取到座位一定时长的使用权限。至此，扫码入座流程结束，如图 4-6 所示。

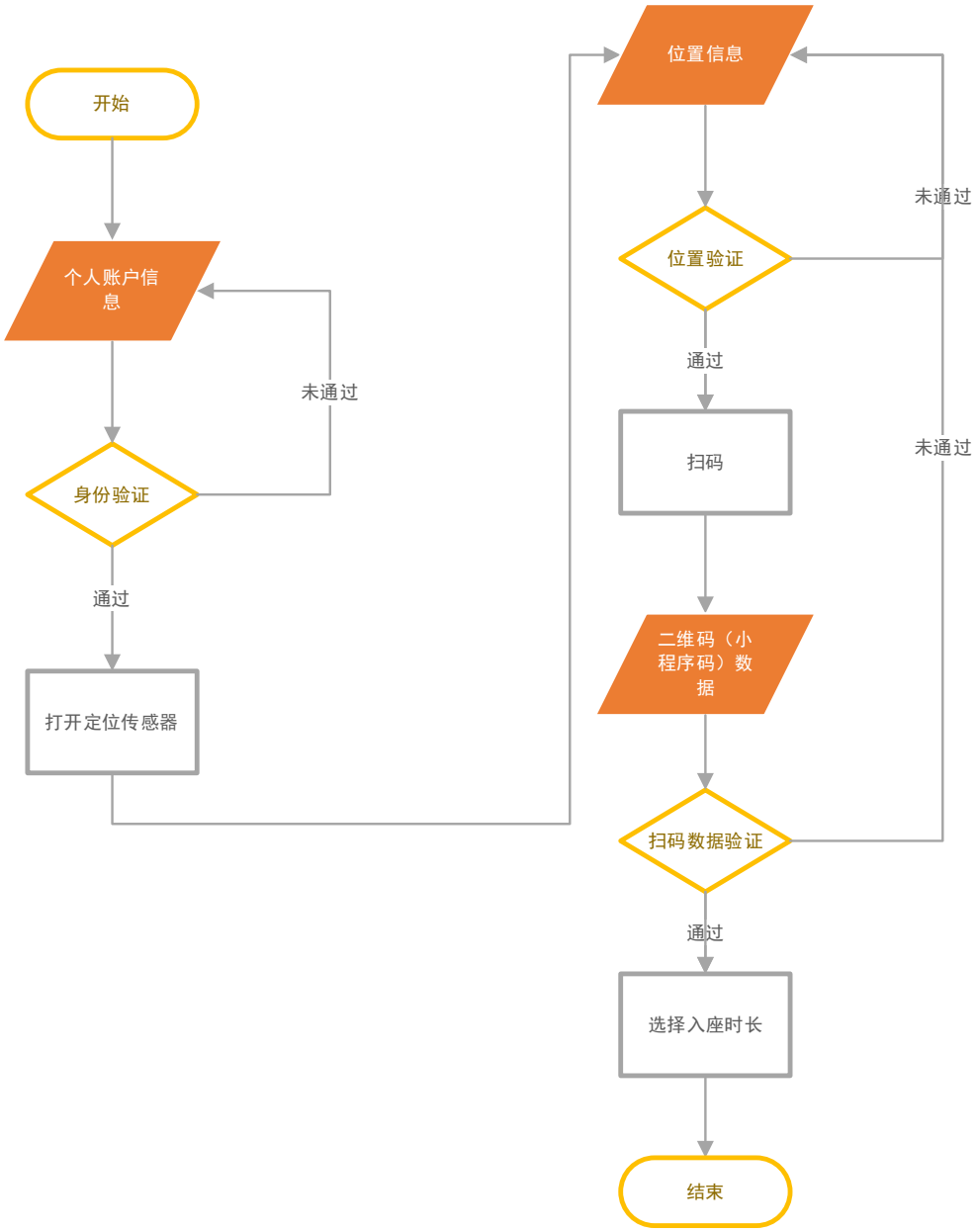


图 4-6　扫码入座流程图

4.3.5 学习历史模块

每次与图书馆内扫码使用座位学习后，会在系统数据库中增添一条关于此次学习的历史记录。记录具体包括学习开始时间、学习结束时间、学习类型等。每个学生可以查看管理自己的学习记录，如图 4-7 所示

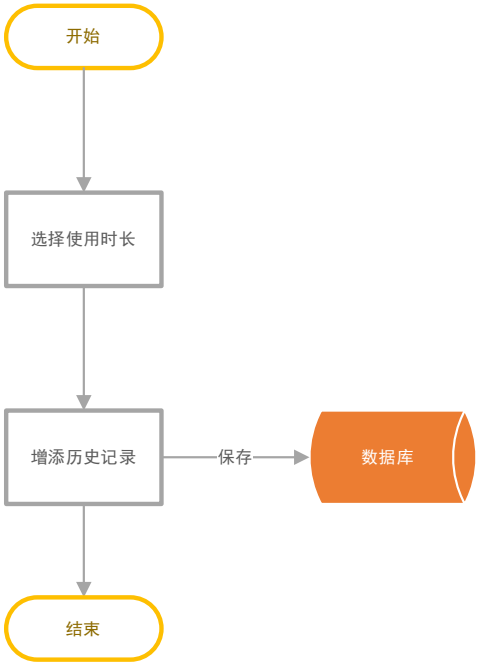


图 4-7　学习历史增添流程图

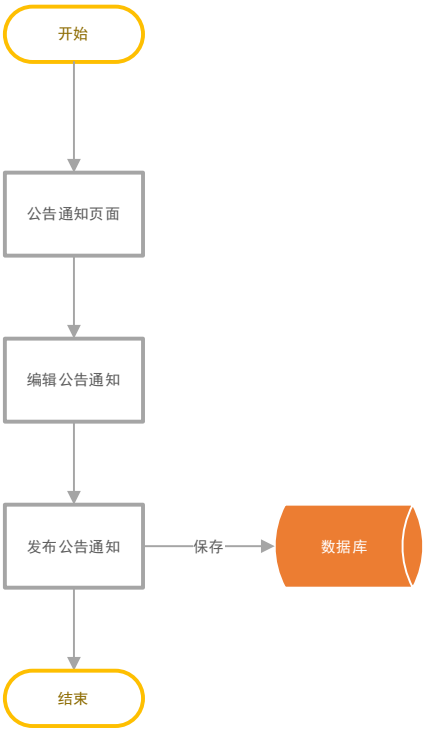


图 4-8　发布公告流程图

4.3.6 通知与公告模块

图书馆管理员可以在微信小程序中发布通知和公告，通知和公告帮助高校图书馆传达图书馆的临时闭关和开馆时间安排、图书馆组织的活动以及学校关于图书馆的相关政策。通知与公告模块帮助图书馆座位管理相关的其他日常工作。图书馆管理员发布通知公告流程如图 4-8 所示。

4.4 数据库设计

4.4.1 关系模型

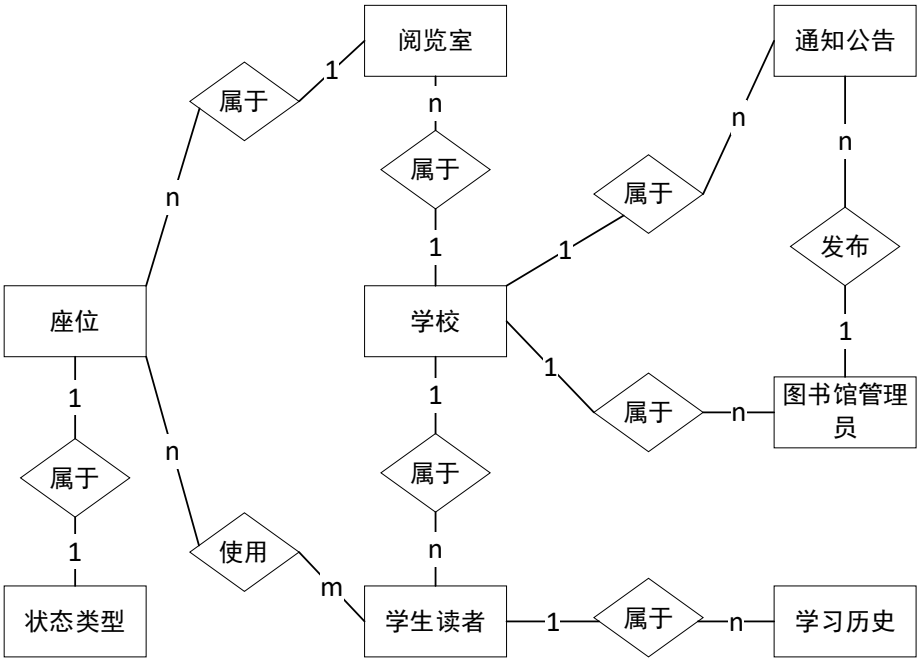


图 4-9　数据库实体-联系（E-R）图

学校、学生读者、学习历史、通知公告、阅览室、图书馆管理员、座位、状态类型八个实体之间的关系如图 4-所示。学生实体属性图如图 4-10 所示，管理员实体属性图如图 4-11 所示，图书馆实体属性图如图 4-12 所示，学习历史实体属性图如图 4-13 所示，阅览室实体属性图如图 4-14 所示，座位实体属性如图 4-15 所示，状态类型实体属性图如图 4-16 所示，通知公告实体属性图如图 4-17 所示。

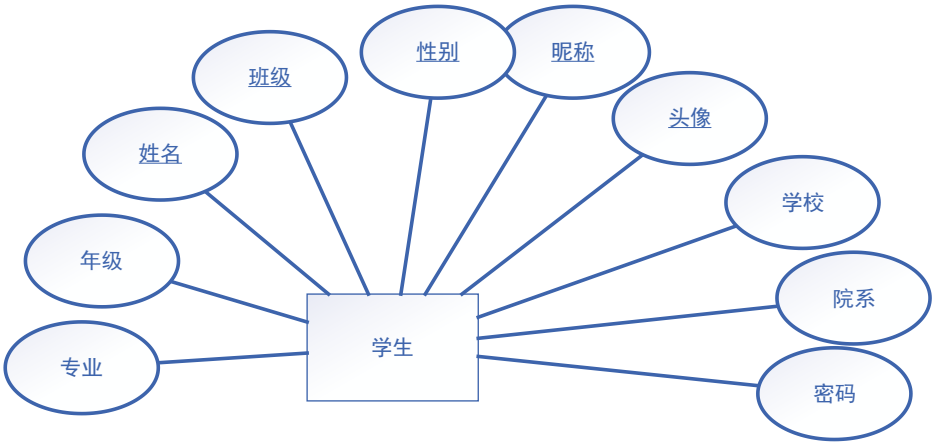


图 4-10　学生实体属性图

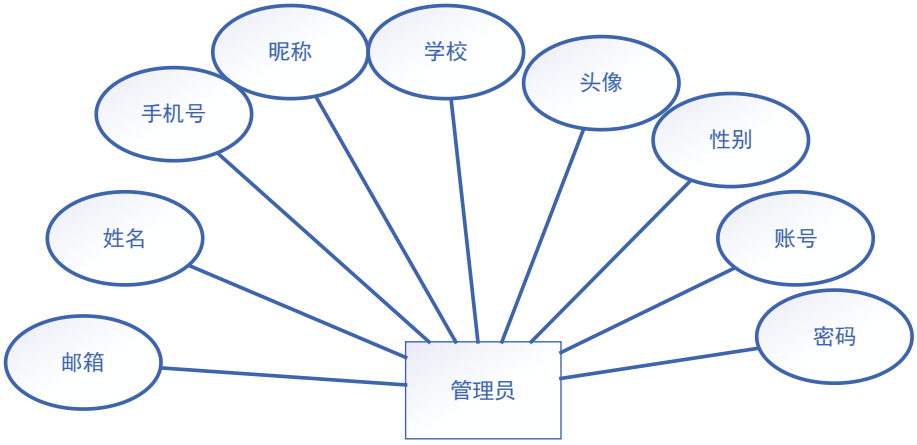


图 4-11　管理员实体属性图

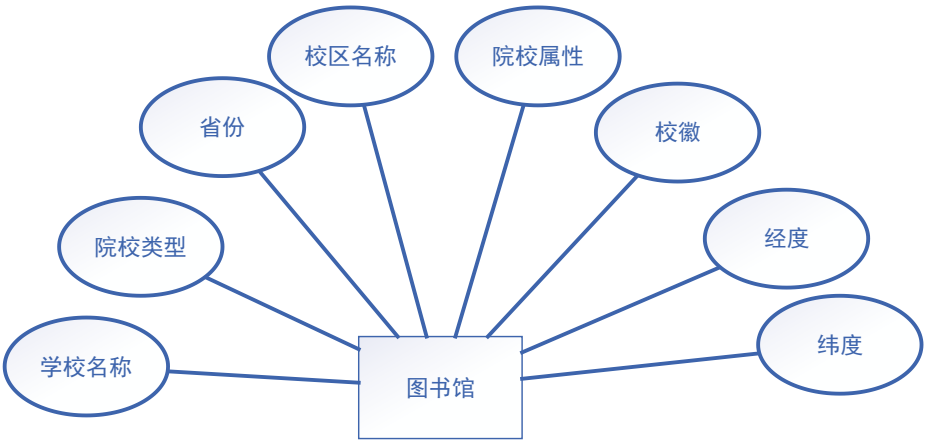


图 4-12　图书馆实体属性图

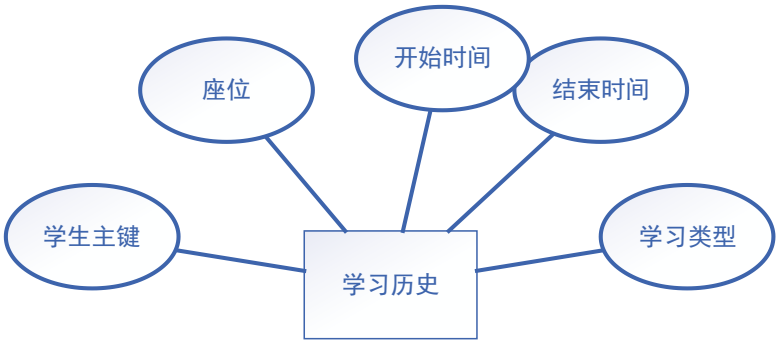


图 4-13　学习历史实体属性图

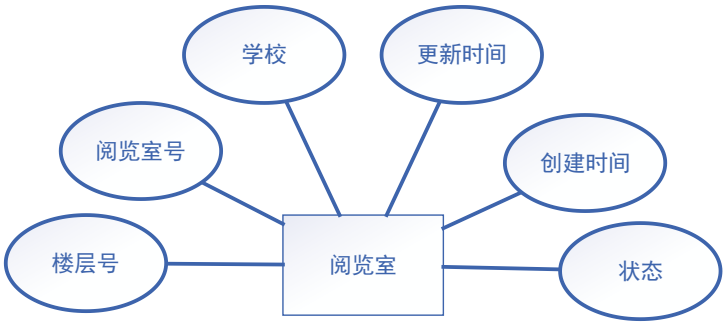


图 4-14　阅览室实体属性图

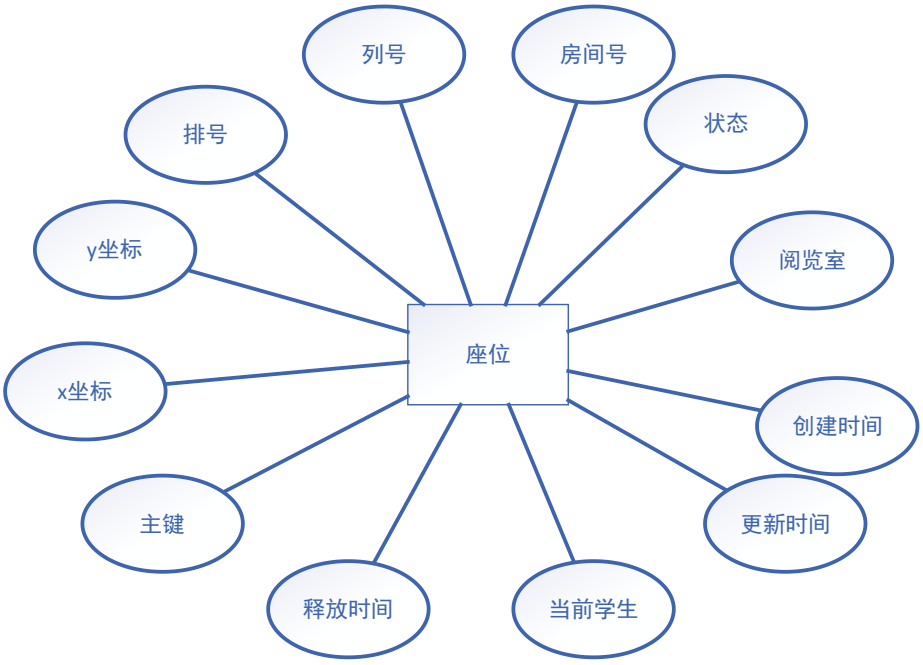


图 4-15　座位实体属性图

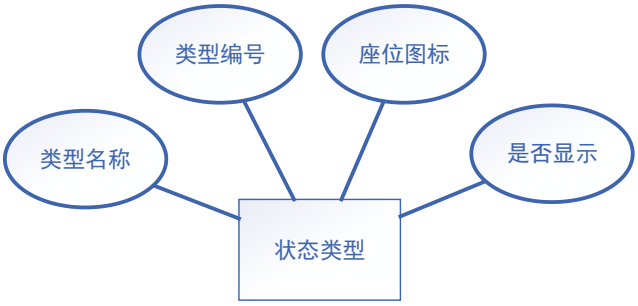


图 4-16　状态类型实体属性图

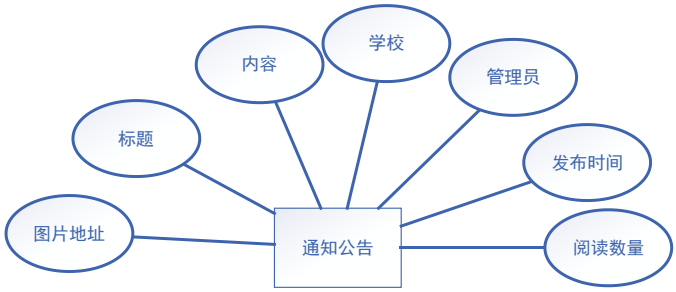


图 4-17　通知公告实体属性图

4.4.2 数据表格

表 4-1　学生用户信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	openid	微信主键	varchar			✓	
3	pwd	密码	varchar			✓	
4	url	头像地址	varchar			✓	
5	name	学生姓名	varchar			✓	
6	nickname	昵称	varchar			✓	
7	sex	性别	enum				保密
8	s_id	学校	int		✓	✓	
9	department	院系	varchar			✓	
10	professional	专业	varchar			✓	
11	class_name	班级	varchar			✓	
12	grade	年级	int			✓	
13	status	是否违规	int				0
14	flag	是否删除	tinyint				0

系统需要对学生用户信息进行存储绑定，学生个人姓名、学校、院系、专业、年级、性别、头像等信息数据库中字段名称、中文含义、字段类型、是否主外键、能否为空以及默认值设计如表 4-1 所示。

表 4-2　管理员用户信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	openid	微信主键	varchar			✓	
3	pwd	密码	varchar			✓	
4	url	头像地址	varchar			✓	
5	name	姓名	varchar			✓	
6	nickname	昵称	varchar			✓	
7	sex	性别	enum				未知
8	s_id	学校	int		✓	✓	
9	phone	手机号	varchar			✓	
10	email	邮箱	varchar				0
11	flag	是否删除	tinyint				0

表 4-3　高校图书馆信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	school_name	学校名称	varchar			✓	
3	campuse_name	校区名称	varchar			✓	
4	province	所在省份	varchar			✓	
5	type	院校类型	varchar			✓	
6	attributes	院校属性	varchar			✓	
7	badge	校徽地址	varchar			✓	
8	longitude	图书馆经度	double			✓	
9	latitude	图书馆纬度	double			✓	0
10	flag	是否删除	tinyint				0

因为学生用户与管理员用户之间数据存储存在区别，系统数据库设计管理员用户信息表如表 4-2 所示，管理员需要存储姓名、昵称、性别、学校、手机号、邮箱等个人信息。

本系统面向各大高校，针对各大高校图书馆的信息数据库存储设计如表 4-3 所示，高校需要存储的信息有学校名称、校区名称、学校的所在省份、学校的类

型、院校属性、校徽存储地址、学校图书馆经度坐标、学校图书馆纬度坐标等信息。

通知与公告模块数据库存储设计主要所需的信息有公告通知的头图地址、通知标题、通知的内容、通知的所属学校、发布这一通知的管理员、发布通知的时间、通知公告的阅读数量等，具体表所示。

表 4-4　通知与公告信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	imgUrl	头图地址	varchar			✓	
3	title	标题	varchar			✓	
4	content	内容	longtext			✓	
5	school_id	学校	int		✓		
6	librarian_id	发布管理员	int		✓	✓	
7	create_time	发布时间	datetime				
8	num	阅读数量	int			✓	0
9	flag	是否删除	tinyint				0

对学生每次的使用座位情况，系统将进行数据库存储，每次学习记录将存储学生的微信主键、学习使用的座位信息、学习开始时间、学习结束时间、学习类型等，具体如表 4-5 所示。

表 4-5　学习历史信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	openid	微信主键	varchar			✓	
3	seat_id	座位	varchar		✓		
4	start_time	开始时间	datetime			✓	
5	end_time	结束时间	datetime				
6	type	学习类型	varchar		✓		

这里的阅览室抽象为房间，阅览室信息数据库表中包括阅览室的楼层号、阅览室房间号、所属学校、信息更新时间、创建时间、状态等，其中状态表示出阅览室是否可用信息。具体如表 4-6 所示。

基于微信小程序的图书馆座位管理系统的管理的主要实体为图书馆座位，座位信息的数据库存储设计尤为重要。座位信息存储表包括座位的横纵坐标、座位

的排号和列号、阅览室房间号、当前的座位使用状态、创建时间、更新时间等相关信息。另外，座位信息表中，每一个座位当前使用的学生相关信息，座位使用结束后的释放时间，具体如表 4-7 所示。

表 4-6　阅览室（房间）信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	floor_num	楼层号	int			✓	
3	room_num	阅览室号	int		✓	✓	
4	s_id	学校	int		✓		
5	update_time	更新时间	datetime			✓	
6	create_time	创建时间	datetime			✓	
7	status	状态	tinyint			✓	0
8	flag	是否删除	tinyint				0

表 4-7　座位信息表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	x	横坐标	int			✓	
3	y	纵坐标	int			✓	
4	row	排	int			✓	
5	col	列	int			✓	
6	room_num	房间号	int			✓	
7	type	状态	varchar			✓	
8	templete_id	阅览室	int		✓	✓	
9	create_time	创建时间	datetime			✓	
10	update_time	更新时间	datetime			✓	
11	reader_openid	当前学生	varchar			✓	
12	release_time	释放时间	datetime			✓	
13	flag	是否删除	tinyint				0

座位的的状态有多种类型，各大高校图书馆的座位状态类型可能不尽相同，系统将座位的状态类型单独存储到座位状态类型表中，包括类型、类型名称、对应的座位图标地址、是否在小程序上显示图标等，具体如表 4-8 所示。

表 4-8　座位状态类型表

序号	名称	中文含义	字段类型	主键	外键	能否为空	默认值
1	id	唯一主键	int	✓			
2	type	类型	varchar				
3	name	类型名称	varchar		✓	✓	
4	icon	座位图标	varchar			✓	
5	isShow	是否显示	tinyint				1

4.5 本章小结

本章描述了系统的设计原则，概述了系统的总体设计，然后对系统的用户管理模块、阅览室管理模块、座位管理模块、扫码入座模块、学习历史模块、通知与公告模块功能模块设计进行详细描述。最后对系统的数据库设计进行描述，包括数据库的数据表格以及数据关系模型。

第五章　系统实现及应用效果

5.1 系统运行环境

系统后台部署在阿里云服务器，处理器为 2 核，4G 内存，40G 硬盘，服务器操作系统为 CentOS 8.0 64 位版本，WEB 服务器为 Apache Tomcat/9.0.30，网络带宽为 5Mbps。微信小程序由微信团队审核后部署在微信服务器，前端小程序页面可在 IOS、Android、PC 微信端等多平台多系统运行。

5.2 系统实现关键技术

5.2.1 系统前后端技术

系统采用面向对象的方法开发整个软件系统，系统主要分为微信小程序端、网页管理端、后台三个部分。这三个部分采用不同的开发技术。

1） 微信小程序端

小程序前端的开发主要采用微信开发者工具，运用 JavaScript 开发语言，开发整个微信小程序的架构页面。

2） 网页管理端

本系统使用 JetBrains WebStorm 开发工具，Vue.js 框架搭建网页管理端的前端页面，主要使用的技术有 HTML、CSS、JavaScript 等开发技术。

3） 系统后台

系统后台使用 IntelliJ IDEA 开发工具，主要采用的技术有 Spring Boot 框架、MyBatis、Redis 缓存技术搭建后台。数据库使用 MySQL。

5.2.2 座位布局算法

由于每个用户的微信小程序客户端的分辨率的不同，如何解决在不同屏幕分辨率下友好的展示阅览室座位布局是小程序端需要解决的问题。算法步骤如下：

步骤 1　循环遍历座位列表数据，比较每个座位 X 坐标，得出最大轴 X 轴坐标；

步骤 2　循环遍历座位列表数据，比较每个座位 Y 坐标，得出最大轴 Y 轴坐标；

步骤 3　根据每个座位图标像素大小计算出未进行比例变换时全部座位在 X 轴方向所占宽度和 Y 轴方向所宽度；

步骤 4 步骤三得出的宽度与相应的微信小程序用户客户端屏幕分辨率大小进行比较，如果大于用户客户端屏幕分辨率，执行步骤五；

步骤 5 座位大小进行一定比例的缩小，使得不同阅览室座位布局中座位在不同分辨率屏幕完全友好的展示出来。

具体代码实现过程参考附录微信小程序源码。

5.3 系统运行效果

5.3.1 阅览室管理页面

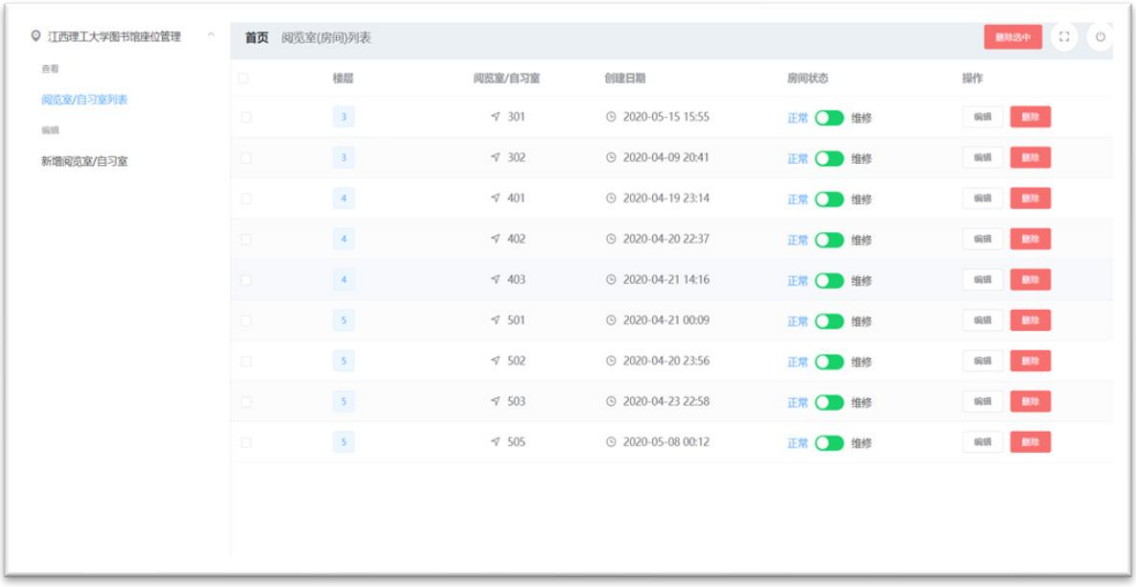


图 5-1 管理员阅览室管理页面

如图 5-1 所示阅览室（自习室）管理页面，页面主要展示楼层、阅览室号、阅览室创建时间、阅览室状态等信息列表，图书馆管理员可以点击“编辑”、“删除”按钮对阅览室进行操作。另外点击列表前的多选框可以对阅览室进行批量删除操作。点击“编辑”按钮页面跳转到阅览室信息页面（如图 5-2 所示），管理员可以自定义阅览室的布局坐标，修改或添加阅览室的楼层号、阅览室号等信息。修改或添加完成后点击确认完成按钮，阅览室信息添加或修改完成。

5.3.2 座位管理页面

图书馆管理员可以在电脑端打开创建座位布局页面，如图 5-3 所示。在这个页面管理员用户可以根据图书馆的座位实际摆放情况，自定义创建座位，座位的状态也可根据实际情况选择普通座位、维修座位和过道座位状态。

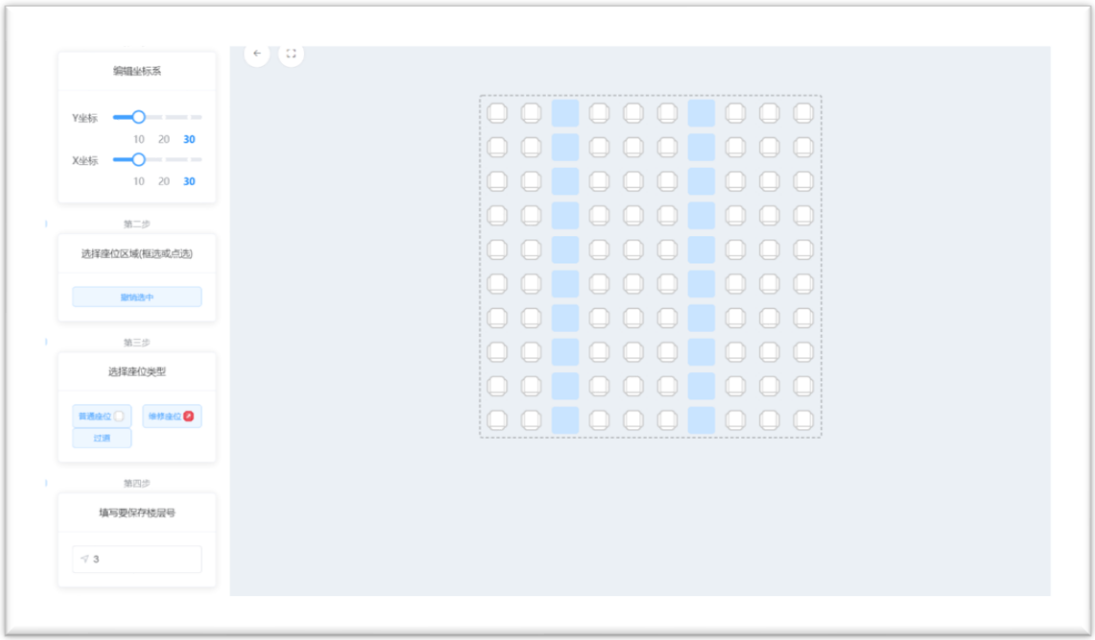


图 5-2　管理员修改阅览室信息页面

(a) 座位坐标系类型编辑页面　　(b) 座位楼层号、房间号编辑页面

图 5-3　管理员创建座位布局页面

在小程序端，图书馆管理员可以查看阅览室内座位使用情况，如图 5-4（a）所示。管理员点击“有人”状态座位可查看使用当前座位的学生个人信息。点击

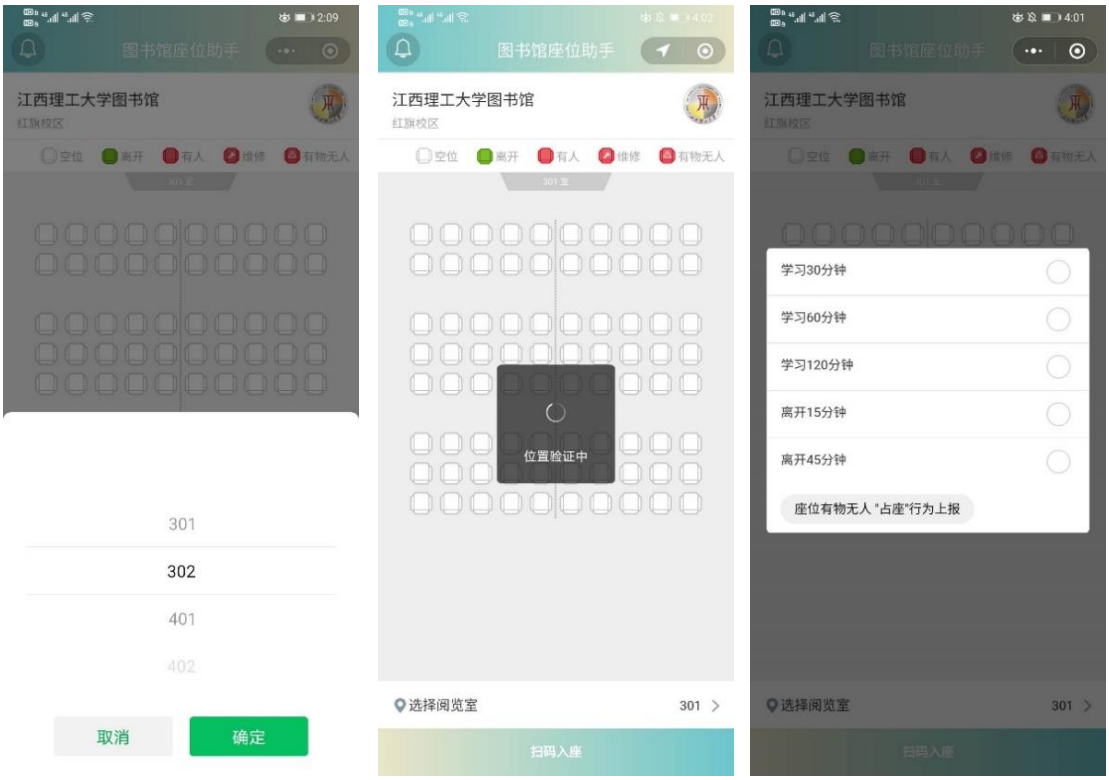
“有物无人”状态的座位可以进行点击“空位”更新座位状态，如图 5-4（b）。

(a) 座位信息展示页面　　(b) 座位当前使用学生信息页面

图 5-4 小程序座位信息页面

5.3.3 扫码入座页面

扫码入座主要分为三步，学生用户首先点击首页下方选择器组件选择阅览室（如图 5-5（a）所示），然后进行位置验证（如图 5-5（b）所示），页面提示“位置中”提示框，位置信息验证通过后，调起微信小程序扫码能力页面（如图 5-6（a）所示），扫描图书馆内座位对应的座位二维码（如图 5-6（b）所示）后，位置信息、个人信息、扫码座位信息验证全部通过后，页面弹出选择时间弹窗，有“学习 30 分钟”、“学习 60 分钟”、“学习 120 分钟”、“离开 15 分钟”、“离开 45 分钟”等按钮，选择学习的时间（如图 5-5（c）所示），扫码入座流程完成。页面更新座位状态，首页选中的阅览室将更新为扫码的座位所在的阅览室。底部扫码入座按钮文字由“扫码入座”转变为学生用户选择的时间倒计时，剩余学习时间将动态的刷新，刷新频率为一秒。其中选择阅览室采用微信选择器组件展示，位置信息验证时展示加载位置验证使用微信弹窗组件。扫码入座的全过程在小程序首页以弹窗的形式展示，点击之后在页面上弹窗消失，页面响应流畅，无跳转页面而产生的短暂留白，用户体验良好。



(a)选择阅览室页面　(b)位置验证页面　(c)选择学习时长页面

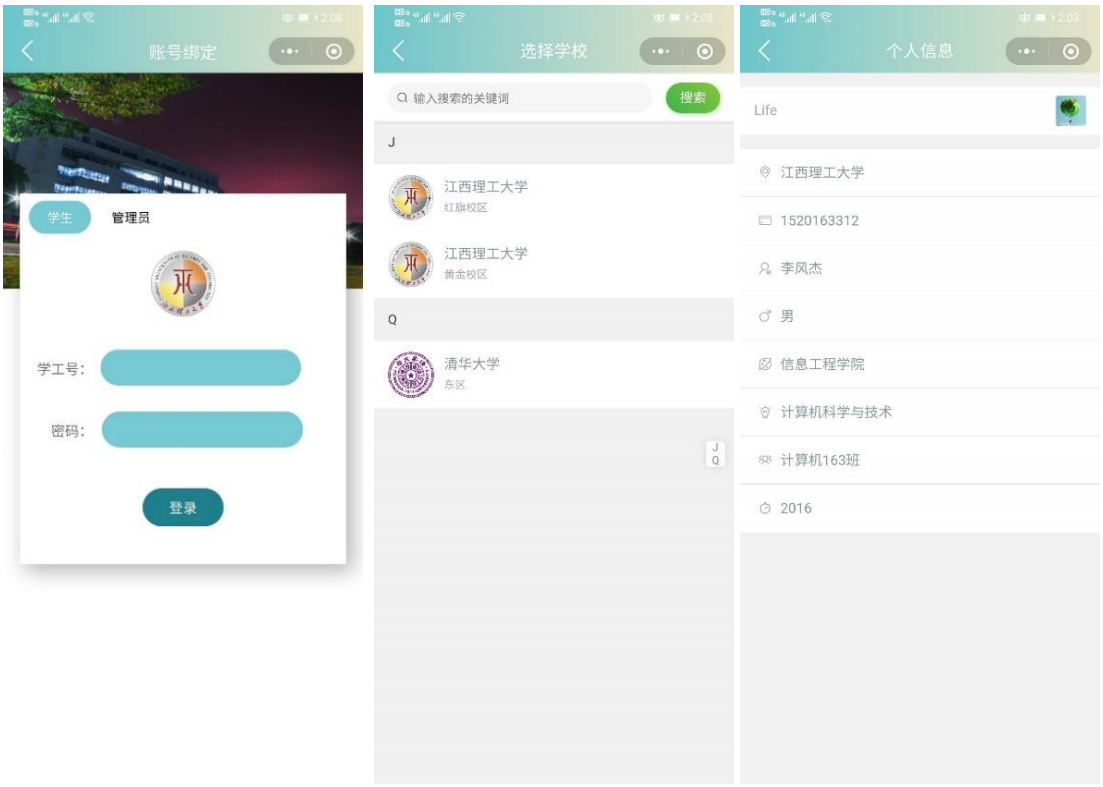
图 5-5 入座身份验证流程页面

(a)扫码页面　(b)座位二维码

图 5-6　学生扫码页面及二维码

5.3.4 用户管理页面

用户管理模块展示部分主要有用户登录页面，学校列表页面，个人绑定的信息页面。如图 5-7 所示。其中用户主要分为学生读者用户和图书馆管理员用户，在登录页面，点击“学生”或“管理员”按钮可以切换用户登录身份,点击“学生”按钮后，输入学工号、密码登录后将是学生用户状态。点击“管理员”按钮后，输入学工号、密码登录后将是管理员用户状态。如图 5-7（a）。如图 5-7（b）所示，学校列表页面中学校数据根据学校名称的首字母顺序分组排序展示，展示学校的校徽、学校名称、学校校区等信息。如图 5-7（c）所示，个人中心页面展示了学生用户的微信昵称、微信头像、姓名、学号、学院、专业、班级、性别等个人信息。在页面上方为一个搜索框和搜索按钮，输入学校名称，点击搜索按钮，下方列表将显示出搜索结果。如果微信昵称、微信头像未获取时，页面上方的微信昵称、微信头像处于预览状态，微信昵称显示“点击获取昵称”，微信头像显示黑色矩形预览块。点击预览图获取用户的微信昵称、微信头像，并将获取的数据展示在小程序页面上。



（a）用户登录页面　　（b）学校列表页面　　（c）用户个人信息页面

图 5-7 学生信息绑定流程页面

5.3.5 学习历史页面

学习历史页面主要展示学生的学习历史信息列表，以时间轴的形式展示。展

示学生每次扫描图书馆座位上二维码使用座位学习后的学习开始时间、学习的结束时间，在时间轴左侧为每次的学习类型，包括“学习”“短暂离开”多种类型，如图 5-8 所示。

5.3.6 编辑公告页面

图书馆管理员可以通过微信小程序编辑公告并发布通知公告，在编辑公告页面，顶部可以单击选择公告通知的头图，下面依次是公告的标题、公告的内容。在页面的底部是一个“发布”按钮。如图 5-9 所示。点击发布按钮，页面会检测公告标题的长度、公告内容的长度，如果公告标题的长度不为空并且公告内容的长度也不为空，编辑公告页面弹出“是否确认发布？”的确认弹窗，如果点击确认，页面将编辑的数据提交给系统后台处理。后台处理成功后，发布公告成功，页面弹出“发布成功”的提示，并且编辑公告页面的公告的标题、公告的内容、发布公告通知的头图被清空，当前编辑公告页面将被关闭。发布成功的公告将展示在公告列表页面中；如果点击取消，发布确认提示弹窗消失，可以继续编辑公告。

5.3.7 通知与公告页面

图 5-8 学生学习历史页面

如图 5-9（a）所示，在通知公告信息流页面可以查看学校的全部通知公告，点击进入公告的详情页面，详情页以富文本的形式展示，如图 5-9（b）所示。

图 5-9 编辑公告页面

（a）通知公告信息流页面

（b）通知公告详情页面

图 5-9 通知公告页面

5.4 系统测试

系统网页管理端采用了目前主流的浏览器（谷歌 Chrome 浏览器、火狐浏览器、QQ 浏览器、360 浏览器等）进行页面和功能测试。

系统微信小程序端分别使用 iPhone 5、iPhone 6/7/8、iPhone 6/7/8 Plus、iPhone XR、Nexus 5、Nexus 5X、Nexus 6、iPad、iPad Pro 等不同分辨率机型测试功能及页面。测试结果如表 5-1 至表 5-8 所示：

表 5-1　用户管理模块测试结果表

序号	测试功能	测试过程	测试结果
1	学生信息绑定	打开微信小程序信息绑定页面，选择江西理工大学红旗校区，输入测试账号密码，点击绑定	绑定成功
2	管理员信息绑定	打开微信小程序信息绑定页面，选择江西理工大学红旗校区，输入管理员测试账号密码，点击绑定	绑定成功
3	查看个人信息	绑定成功后重新进入信息绑定页面显示	个人信息显示成功

表 5-2　阅览室管理模块测试结果表

序号	测试功能	测试过程	测试结果
1	添加阅览室	打开管理后台网页，登录管理员账号，点击添加阅览室按钮，填写楼层号：3，阅览室号：301,点击添加按钮	添加成功
2	删除阅览室	打开管理后台网页，登录管理员账号,点击删除 301 阅览室按钮	删除成功
3	修改阅览室信息	打开管理后台网页，登录管理员账号,点击修改 301 阅览室按钮，将 301 修改为 302，点击提交	修改成功

表 5-3　座位管理模块测试结果表

序号	测试功能	测试过程	测试结果
1	添加座位	打开管理后台网页，登录管理员账号，点击编辑阅览室按钮，拖动添加座位	添加成功
2	删除座位	打开管理后台网页，点击编辑阅览室按钮，点击座位为空座	删除成功
3	修改座位状态	打开管理后台网页，点击编辑阅览室按钮，点击空位座位为维修状态	修改成功

表 5-4　扫码入座模块与学习历史模块测试结果表

序号	测试功能	测试过程	测试结果
1	扫码入座	绑定并登录微信小程序后，点击小程序上扫码入座按钮，验证位置信息后，扫描测试二维码，选择 30 分钟	入座成功，座位状态发生改变
2	添加学习历史	扫码入座成功后，点击打开历史信息页面	历史信息已添加
3	占座上报	绑定并登录微信小程序后，点击小程序上扫码入座按钮，验证位置信息后，扫描测试二维码，点击“座位有物无人 ‘占座’行为上报”按钮	上报成功，座位状态发生改变

表 5-5　通知与公告模块测试结果表

序号	测试功能	测试过程	测试结果
1	发布通知公告	管理员绑定并登录微信小程序后，编辑测试公告，点击发布	发布成功
2	查看通知与公告	打开小程序通知与公告列表，点击打开详情页面	数据显示成功

表 5-6　管理员扫码查看座位信息测试结果表

序号	测试功能	测试过程	测试结果
1	扫码查看座位信息	绑定并登录微信小程序后，点击小程序上扫码查看座位信息按钮，验证位置信息后，扫描测试二维码	查看成功，显示出使用座位学生的个人信息

表 5-7　管理员清空座位信息测试结果表

序号	测试功能	测试过程	测试结果
1	管理员清空座位信息	绑定并登录微信小程序后，点击小程序上扫码查看座位信息按钮，点击清空座位按钮	清空成功，座位状态转变为“空位”状态

表 5-8　超时后座位状态更新测试结果表

序号	测试功能	测试过程	测试结果
1	超时后座位状态更新	绑定并登录微信小程序后，点击小程序上扫码入座按钮，验证位置信息后，扫描测试二维码，选择 30 分钟，等待 30 分钟	座位状态由有人状态转化为空位状态

5.5 本章小结

本章主要描述了系统的实现以及应用效果。首先讲述了系统的运行环境，然后描述了系统的实现方法，接着展示和详细描述了系统不同模块的运行效果，对页面展示效果和页面联系进行了详细描述。在章节末尾，对系统所有功能模块进行系统测试，详细列出测试功能、测试功能和测试结果。

第六章　总结与展望

本课题基于微信小程序的图书馆座位管理系统，针对各大高校面临的图书馆座位硬件资源不足的情况，提出软件系统管理解决方案。系统采用微信小程序客户端和网页的形式向学生和图书馆管理员提供图书馆座位的实时状态。系统在一定程度上讲，降低了高校图书馆座位资源的浪费，提升了现有的座位的使用效率。提高了图书馆在一定时间段的容纳使用座位学习的学生总数量。总的来说，本文完成了基于微信小程序的图书馆座位管理系统的开发工作，主要成果如下：

1）系统采用微信小程序客户端，学生通过扫码的形式将图书馆的座位使用情况上传至服务器。系统将图书馆座位使用情况数据汇总。

2）本系统降低了图书馆管理员的日常巡查工作强度。

3）本系统降低了学生寻找图书馆空座位的时间。

因为时间、服务器硬件资源等原因的限制，基于微信的图书馆座位管理系统还存在可以改进和优化的空间，具体可以从以下几个方面进行改进优化：

1）微信小程序前端界面以及性能方面可以进一步优化，更加符合高校学生的使用习惯。

2）管理员管理网页端可以进一步增加图书馆管理相关业务功能，进一步提升图书馆相关工作的效率。

3）系统后台方面可以进一步对接口性能、响应时间、数据库存储等进一步优化，在服务器资源一定的情况下，服务更多高校图书馆学生。

附　录

1 后台源码

1.1 阅览室管理模块源码

```
@RestController
public class mainController {
    @Autowired
    private SeatService seatService;
    @Autowired
    private SeatTempleteService seatTempleteService;

    @RequestMapping(value = "/insertSeat", method = RequestMethod.POST)
    @ResponseBody
    public Object insertSeat(@RequestBody Seats seats) {
        if (seats != null) {
            List<SeatObj> processedList = new ArrayList<>();
            List<SeatObj> seatList = seats.getSeats();
            List<Integer> yList = seatList.stream()
                    .map(SeatObj::getY)
                    .sorted()
                    .distinct()
                    .collect(Collectors.toList());
            int row = 1;
            for (Integer y : yList) {
                int col = 1;
                List<SeatObj> rowList = seatList.stream().filter(t ->
t.getY().equals(y)).collect(Collectors.toList());
                rowList =
rowList.stream().sorted(Comparator.comparing(SeatObj::getX)).collect(Collectors.to
List());
                for (SeatObj seat : rowList) {
                    seat.setRow(row + "");
                    seat.setCol(col + "");
                    col++;
                }
                row++;
                processedList.addAll(rowList);
            }
            if (processedList.size() > 0) {
                SeatTemplete seatTemplete = new SeatTemplete();
                seatTemplete.setFloorNum(seats.getTempleteFloorNum());
                seatTemplete.setRoomNum(seats.getTempleteRoomNum());
```

```
                    seatTemplete.setId(seats.getTempleteId());
                    seatTemplete.setFlag(0);
                    seatTemplete.setStatus(false);
                    try {
                        seatService.insertSeatAndReplace(processedList,
seatTemplete);
                        ResponseBean responseBean = new ResponseBean();
                        responseBean.setErrCode(0);
                        responseBean.setErrMsg("成功");
                        return responseBean;
                    } catch (Exception e) {
                        ResponseBean responseBean = new ResponseBean();
                        responseBean.setErrCode(1);
                        responseBean.setErrMsg("失败");
                        return responseBean;
                    }
                }

            }
            return "success";
        }

        @RequestMapping(value = "/findTempleteList", method = RequestMethod.GET)
        @ResponseBody
        public Object findTempleteList(@RequestParam("schoolId")String sId) {
            try {
                List<SeatTemplete>                    list                    =
seatTempleteService.findByTempleteList(sId);
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(0);
                responseBean.setErrMsg("成功");
                responseBean.setData(list);
                return responseBean;
            } catch (Exception e) {
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(1);
                e.printStackTrace();
                return responseBean;
            }
        }

        @RequestMapping(value = "/deleteTemplete", method = RequestMethod.POST)
        @ResponseBody
        public Object deleteTemplete(@RequestBody SeatsIdList idList) {
```

```
            try {
                seatTempleteService.deleteTemplete(idList.getIdList());
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(0);
                responseBean.setErrMsg("成功");
                return responseBean;
            } catch (Exception e) {
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(1);
                responseBean.setErrMsg("失败");
                return responseBean;
            }
        }
@RequestMapping(value="/findSeatListByTempleteId",method=RequestMethod.GE
T)
        @ResponseBody
        public Object findSeatListByTempleteId(@RequestParam(value = "id") Integer id)
{
            try {
                Seats seats = seatService.findSeatListByTempleteId(id);
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(0);
                responseBean.setErrMsg("成功");
                responseBean.setData(seats);
                return responseBean;
            } catch (Exception e) {
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(1);
                responseBean.setErrMsg("失败");
                return responseBean;
            }
        }
}
```

1.2 座位管理模块源码

```
@RestController
public class seatController {
        @Autowired
        private SeatService seatService;
        @Autowired
        private ReaderService readerService;
        @Autowired
        private LibrarianService librarianService;
        @Autowired
```

```
private SeatTypeService seatTypeService;
@Autowired
private SeatTempleteService seatTempleteService;
@RequestMapping(value = "/seatList", method = RequestMethod.POST)
@ResponseBody
public Object findSeatListByTempleteId(
        @RequestParam(value = "id") Integer templeteId,
        @RequestParam(value = "openid") String openid
        ) {
    try {
        Reader reader = readerService.findByOpenid(openid);
        Librarian librarian = librarianService.findByOpenid(openid);
        if(reader == null && librarian == null) {
            ResponseBean responseBean = new ResponseBean();
            responseBean.setErrCode(-1);
            responseBean.setErrMsg("请先绑定学校账号后重试");
            return responseBean;
        }
        SeatsResponseBean seatsResponseBean = new SeatsResponseBean();
        Seats seats = seatService.findSeatListByTempleteId(templeteId);
        seatsResponseBean.setErrCode(0);
        seatsResponseBean.setErrMsg("success");
        String campuses = null;
        String schoolName = null;
        String schoolUrl = null;
        if(reader != null){
            campuses = reader.getSchool().getCampuses();
            schoolName = reader.getSchool().getSchoolName()+"图书馆";
            schoolUrl = reader.getSchool().getUrl();
            seatsResponseBean.setUType("reader");
        }
        else {
            campuses = librarian.getSchool().getCampuses();
            schoolName = librarian.getSchool().getSchoolName();
            schoolUrl = librarian.getSchool().getUrl();
            seatsResponseBean.setUType("librarian");
        }
        seatsResponseBean.setCampuses(campuses);
        seatsResponseBean.setSchoolName(schoolName);
        seatsResponseBean.setSchoolUrl(schoolUrl);
        seatsResponseBean.setRoomNum(seats.getTempleteRoomNum());
        seatsResponseBean.setSeatList(seats.getSeats());
        List<SeatType> seatTypeList = seatTypeService.findAll();
        seatsResponseBean.setSeatTypeList(seatTypeList);
```

```
            return seatsResponseBean;
        } catch (Exception e) {
            ResponseBean responseBean = new ResponseBean();
            responseBean.setErrCode(1);
            e.printStackTrace();
            responseBean.setErrMsg("fail");
            return responseBean;
        }
    }

    @RequestMapping(value = "/roomNumList", method = RequestMethod.GET)
    @ResponseBody
    public Object findTempleteIdList(@RequestParam(value = "openid") String
openid) {
        try {
            Reader reader = readerService.findByOpenid(openid);
            Librarian librarian = librarianService.findByOpenid(openid);
            if(reader == null && librarian == null) {
                ResponseBean responseBean = new ResponseBean();
                responseBean.setErrCode(-1);
                responseBean.setErrMsg("请先绑定学校账号后重试");
                return responseBean;
            }
            Integer schoolId = 0;
            if(reader != null){
                schoolId = reader.getSchool().getId();
            }else if(librarian != null){
                schoolId = librarian.getS_id();
            }
List<SeatTemplete>list=seatTempleteService.findByTempleteIdList(schoolId);
            ResponseBean responseBean = new ResponseBean();
            responseBean.setErrCode(0);
            responseBean.setErrMsg("success");
            responseBean.setData(list);
            return responseBean;
        } catch (Exception e) {
            ResponseBean responseBean = new ResponseBean();
            responseBean.setErrCode(1);
            return responseBean;
        }
    }

    @RequestMapping(value = "/getNowSeatInfo", method = RequestMethod.GET)
    @ResponseBody
```

```
public Object getNowSeatInfo(
        @RequestParam(value = "seatId") Integer id,
        @RequestParam(value = "openid") String openid
        ){
    Librarian librarian = librarianService.findByOpenid(openid);
    if(librarian == null){
        return null;
    }
    try {
        SeatNow nowSeatInfo = seatService.getNowSeatInfo(id);
        if(nowSeatInfo == null){
            return "空位";
        }
        return seatService.getNowSeatInfo(id);
    }catch (Exception e){
        return null;
    }

}
@RequestMapping(value = "/reportSeat",method = RequestMethod.GET)
public Object reportSeat(
        @RequestParam(value = "openid") String openid,
        @RequestParam(value = "code") String code
){

    ResponseBean responseBean = new ResponseBean();
    Reader reader = readerService.findByOpenid(openid);
    if(reader == null) {
        responseBean.setErrCode(-1);
        responseBean.setErrMsg("请先绑定学校账号后重试");
        return responseBean;
    }
    Integer id = Integer.parseInt(code);
    // 先判断座位使用情况 如果有人使用中，无法抢占
    SeatObj seatIsUse = seatService.checkSeatNowCanUse(id);
    if(seatIsUse == null){
        responseBean.setErrCode(-1);
        responseBean.setErrMsg("当前座位有人正在使用中");
        return responseBean;
    }
    String type = "5";
    try{
        seatService.updateSeatType(id,type);
        responseBean.setData(0);
```

```
            responseBean.setErrMsg("success");
        }catch (Exception e){
            responseBean.setErrMsg("fail");
            responseBean.setErrCode(-1);
        }
        return responseBean;
    }

    @RequestMapping(value = "/clearSeat",method = RequestMethod.GET)
    public Object clearSeat(
            @RequestParam(value = "openid") String openid,
            @RequestParam(value = "seatId") Integer seatId
    ){
        ResponseBean responseBean = new ResponseBean();
        Librarian librarian = librarianService.findByOpenid(openid);
        if(librarian == null) {
            responseBean.setErrCode(-1);
            responseBean.setErrMsg("请先绑定学校账号后重试");
            return responseBean;
        }
        try{
            seatService.clearSeat(seatId);
            responseBean.setErrCode(0);
            responseBean.setData(0);
            responseBean.setErrMsg("success");
        }catch (Exception e){
            responseBean.setErrMsg("fail");
            responseBean.setErrCode(-1);
        }
        return responseBean;
    }
}
```

2 微信小程序源码

```
InitseatList(that) {
      let result = jsonData.dataList
      let durationList = jsonData.durationList
      let seatList = that.prosessSeatList(result);
      //计算 X 和 Y 坐标最大值
      that.prosessMaxSeat(seatList);
      //计算左侧座位栏的数组
      that.seatToolArr()
      //按每排生成座位数组对象
      that.creatSeatMap()
      //确认最佳坐标座位
      that.creatBestSeat()
      this.setData({
        seatList: seatList,
        schoolName: result.schoolName,
        campuses: result.campuses,
        roomNum: result.roomNum,
        schoolUrl: result.schoolUrl,
        seatList: seatList,
        seatTypeList: result.seatTypeList,
        durationList: durationList
      })
  }

prosessMaxSeat: function (value) {
      let seatList = value
      let maxY = 0;
      for (let i = 0; i < seatList.length; i++) {
        let tempY = seatList[i].x;
        if (parseInt(tempY) > parseInt(maxY)) {
          maxY = tempY;
        }
      }
      let maxX = 0;
      for (var i = 0; i < seatList.length; i++) {
        var tempX = seatList[i].y;
        if (parseInt(tempX) > parseInt(maxX)) {
          maxX = tempX;
        }
      }
      let seatRealWidth = parseInt(maxX) * 70 * this.data.rpxToPx
      let seatRealheight = parseInt(maxY) * 70 * this.data.rpxToPx
```

```
    let seatScale = 1;
    let seatScaleX = 1;
    let seatScaleY = 1;
    let seatAreaWidth = 630 * this.data.rpxToPx
    let seatAreaHeight = this.data.seatArea - 200 * this.data.rpxToPx
    if (seatRealWidth > seatAreaWidth) {
      seatScaleX = seatAreaWidth / seatRealWidth
    }
    if (seatRealheight > seatAreaHeight) {
      seatScaleY = seatAreaHeight / seatRealheight
    }
    if (seatScaleX < 1 || seatScaleY < 1) {
      seatScale = seatScaleX < seatScaleY ? seatScaleX : seatScaleY
    }
    this.setData({
      maxY: parseInt(maxY),
      maxX: parseInt(maxX),
      seatScale: seatScale,
      seatScaleHeight: seatScale * 70 * this.data.rpxToPx
    });
  }
  // 座位左边栏的数组
  seatToolArr: function () {
    let seatToolArr = []
    let yMax = this.data.maxY
    let seatList = this.data.seatList
    for (let i = 1; i <= yMax; i++) {
      let el = ''
      for (let j = 0; j < seatList.length; j++) {
        if (parseInt(seatList[j].x) === i) {
          el = seatList[j].row
        }
      }
      seatToolArr.push(el)
    }
    this.setData({
      seatToolArr: seatToolArr
    })
  }
```

参考文献

[1] 国家统计局中国. 2019中国统计年鉴[M]. 北京:中国统计出版社, 2019.677-697.

[2] 何宇蝶. 当代大学生考研率上升的原因[J]. 管理观察, 2019(34):137-138.

[3] 沈慧, 吴仪, 徐慧雯, 等. 基于小程序的图书馆座位预约系统设计与实现[J]. 电脑知识与技术, 2019(12):70-71.

[4] 陆有丽, 邓凯航, 李雯婧, 等. 基于微信小程序的图书馆座位预约系统的设计与实现[J]. 湖南理工学院学报(自然科学版), 2020,33(01):29-33.

[5] 胡念恩. 基于STC90C52单片机的图书馆座位管理系统[J]. 山东工业技术, 2018(11):134.

[6] 张静端. 基于GIS技术的高校图书馆座位管理系统[J]. 东华大学学报(自然科学版), 2016,42(02):242-247.

[7] 杜波. 基于单片机及CAN技术的图书馆自习室座位管理系统的实现[J]. 情报探索, 2008(03):62-63.

[8] 熊双, 方文饶. 基于压力传感的图书馆座位管理系统[J]. 科技创新导报, 2013(12):214.

[9] 叶松涛, 毕蓉蓉. 基于Android平台的图书馆座位管理系统界面设计与实现[J]. 电子技术与软件工程, 2014(23):75-76.

[10] 熊玉涛. 基于SSH框架的图书馆自习室管理系统的设计与实现[D]. 江西财经大学, 2019.

[11] 李红娣. 图书馆移动座位管理信息系统设计与实现[D]. 江西财经大学, 2018.

[12] 伍叶霖. 基于微信的图书馆座位管理系统的设计与实现[D]. 湖南大学, 2018.

[13] 于俊丽. 高校图书馆应用微信小程序的实践与展望[J]. 出版广角, 2018(12):73-75.

[14] 张毅. 基于微信小程序的图书馆座位管理系统[J]. 新世纪图书馆, 2019(8):62-65.

[15] 吕宇琛. SpringBoot框架在web应用开发中的探讨[J]. 科技创新导报, 2018,15(08):168-173.

[16] 李文杰. 基于SSM框架开发平台的教学案例式实践[J]. 四川水泥, 2019(9):268.

[17] 郝佳. Spring源码深度解析[M]. 北京:人民邮电出版社, 2019.

[18] QST青软实训. Java EE轻量级框架应用与开发[M]. 北京:清华大学出版社, 2016.

[19] 颜治平. 基于SpringBoot和Vue框架的教代会提案系统的设计与实现[J]. 科技创新与应用, 2020(03):91-93.

[20] 等吕云翔. 小程序,大未来[M]. 北京:电子工业出版社, 2018.

[21] 廖海玲, 何明昌, 李梅, 等. Oray上的小程序教学辅助平台设计与实现[J]. 电脑知识与技

术, 2019(19):90-92.

[22] 董桂娟. 浅谈高校图书馆座位资源紧缺现象及对策[J]. 吉林化工学院学报, 2011(8):66-67.

[23] 王维秋, 刘春丽. 基于人脸识别技术的我国图书馆智慧服务功能设计与模式构建[J]. 图书馆学研究, 2018(18):44-50.

[24] 李建莎, 谢景卫, 张敬平, 等. 图书馆空座检测及统计综合系统设计[J]. 电子技术, 2018(2):49-52.

[25] 杨雅馨, 郭媛珂, 赵昊, 等. 基于开源硬件下的图书馆桌椅附属设计[J]. 大科技, 2016(31)280-281.

致　谢

不经意间，大学本科四年时光就要结束。在完成论文的最后，我要向帮助我的各位老师、同学最诚挚的谢意。

首先对我的论文指导老师曾鹏程老师表示感谢，谢谢老师在课题的选择、系统设计的设计、论文的撰写等各方面给予的指导和建议。

另外，在此对工作室的老师、同学表示感谢。大学两年多时间，是你们陪伴我一起学习成长。感谢大学中每一位老师和同学，谢谢你们。无论在课堂学习还是大学日常生活上，都给予了我很大的帮助。

基于微信小程序的图书馆座位管理系统

李风杰，曾鹏程

（江西理工大学，信息工程学院，江西 赣州 341000）

摘　要：随着各大高校的不断扩招与考研人数的逐年上升，图书馆座位资源日趋紧张。本文提出基于微信小程序的图书馆座位管理系统，为图书馆座位管理提供高效的解决方案。系统手机客户端使用微信小程序呈现，管理员管理端采用 Vue 和 ElementUI 框架实现。后端采用 Spring Boot、MyBatis 技术和 Redis 缓存技术，使用了 MySQL 数据库存储系统数据。本系统主要模块有阅览室管理模块、座位管理模块、扫码入座模块等。

关键词：微信小程序；图书馆；座位管理

Abstract： With the continuous expansion of enrollment in major universities and the increasing number of graduate students, the library seating resources are becoming increasingly tense. This article proposes a library seat management system based on WeChat applets to provide an efficient solution for library seat management. The mobile phone client of the system is presented by WeChat applet, and the administrator management end is implemented by Vue and ElementUI framework. The back end adopts Spring Boot, MyBatis technology and Redis cache technology, and uses MySQL database to store system data. The main modules of this system include a reading room management module, a seat management module, and a code scanning and seating module.

Keywords： WeChat Mini Programs; library; Seats management

0 前言

根据 2019 中国统计年鉴数据显示，2018 年中国普通本专科在校人数达到 2831 万，相比 2017 年中国普通本专科在校人数 2753 万有所上升[1]。随着各大高校的不断扩招，各大高校图书馆座位资源相继出现座位资源使用紧张的情况。虽然各大高校针对这种情况做出增加座位硬件资源的举措，但是总体上，增加的座位资源有限，并没有改变各大高校图书馆座位资源使用紧张的局面。

近年来考研人数不断攀升，2018 年 12 月 23 日，这一年的考研人数，再一次变成了“历史新高”。从教育部公布的数据来看，2019 年全国考研报名人数达到了史无前例的 290 万人，比 2018 年考研人数增加了 52 万人，增幅达到了 21.8%，考研增加人数和增长率均为近年来最高。近 5 年，考研人数逐年增长，同 2015 年的 165 万人相比增加了 125 万人[2]。图书馆自习室无疑是考研学生复习备考的最佳场所，考研人数的逐年攀升使得各大高校图书馆座位硬件资源更加紧张。另外大学生考证考级热度的不断“升温”，在一定程度上使得高校图书馆座位资源更加紧张。

在高校图书馆座位硬件资源数量有限的情况下，高校图书管中出现不同程度的占座现象，在图书馆常常出现有物无人的座位。这使得需要使用座位的同学无法使用这些有物无人的座位资源，这些座位处于闲置状

态[3]。占座现象过多时，造成图书馆整体可用的座位资源处于一种“假满”状态。这种占座行为浪费了图书馆有限的宝贵的座位资源，降低了图书馆座位资源的使用效率。

1 需求分析

目前各大高校图书馆对座位管理普遍使用传统的管理员巡视的方式，图书馆内座位得不到较为科学的管理，无法使图书馆容纳更多的需要座位学习的学生。管理员无法高效便捷的获取图书馆实时的座位使用情况，面对“占座”行为，只能暂时对有物品而无人使用的座位上的物品进行临时的收纳，这不仅给图书馆管理员带来巨大的工作压力，而且也会对因打热水等不可抗拒原因短暂离开座位的同学物品造成误缴[4]。对物品的收纳也会给高校图书馆方增添物品的管理看护工作，增加因图书馆座位资源不足而造成的学生与图书馆方管理的矛盾。长期而言这种矛盾会影响学生和图书馆管理员双方的工作与学习效率[5]。因此如何解决管理员与学生无法即时获取到图书馆内座位使用情况成为系统需要讨论研究的重要问题。另外如何在高校内图书馆座位硬件资源有限的情况下，提高每个座位的使用效率，以达到服务更多需要在馆内座位学习的学生的问题也是本系统课题的重要需求。

2 概要设计

本系统主要分为用户管理模块、阅览室管理模块、座位管理模块、扫码入座模块、学习历史模块和通知与公告模块等六个大的部分：

1) 用户管理模块

这里的用户主要分为图书馆管理员和学生用户，针对这两类不同的用户类型系统

图 1 系统功能模块结构图

将有不同的功能。对于图书馆管理员主要有管理员个人信息的绑定，PC 网页管理端的登录权限等的划分。学生用户个人信息的绑定，用于使用座位时个人身份验证、学习历史的存储等。

2) 阅览室管理模块

高校图书馆管理员可以根据自己学校的阅览室数量进行创建阅览室，自定义阅览室的信息，包括阅览室房间号、阅览室楼层、创建时间、阅览室状态等信息。图书馆管理员可以对阅览室进行添加、修改、删除操作。

3) 座位管理模块

图书馆管理员可以在 PC 管理网页端自定义阅览室的座位布局以及每一个座位的状态。管理员可以对座位的状态信息、位置信息进行修改操作。在微信小程序端图书馆管理员可以查看每一间阅览室和自习室的座位使用状态、当前使用该座位的学生用户的个人信息。图书馆管理员可以对座位的状态信息进行单独修改。学生用户可以通过微信小程序查看图书馆内每一间阅览室和自习室的座位当前状态，扫描座位二维码验证后使用座位，座位状态发生改变。学生对于图书馆内有物无人的座位可以举报。上报后该座位的状态信息同样发生改变。

4) 扫码入座模块

这个模块主要是让学生扫描二维码后再使用图书馆内的座位资源，在扫码入座前要对学生的身份、位置等信息进行验证。确

保在学生扫描使用座位时，学生处于图书馆内，而且当前未扫描使用其他座位；被扫描的座位未被其他人使用并处于可用的状态。从而做到，一个学生用户在同一时间仅可以使用一个图书馆座位，一个图书馆的座位只能被一个人使用。

3 详细设计

3.1 阅览室管理模块设计

图书馆管理员在电脑网页端以管理员身份登录后，可以通过阅览室（自习室）管理模块对本图书馆的阅览室进行增添、删除、修改操作。阅览室信息包括阅览室所在的楼层号、阅览室或自习室的房间号码、创建日期、上次修改信息日期、阅览室当前状态（正常使用、维修中）等。

3.2 座位管理模块设计

座位管理模块涉及到各大高校图书馆的每一间阅览室和自习室中座位信息的管理。首先，图书馆管理员可以根据自身图书馆阅览室中座位的摆放布局、座位的大小自定义创建座位。管理员可以创建普通座位、正在维修中座位、过道等多种类型的阅览室座位布局。管理员通过电脑浏览器打开 Web 网页可视化点击创建，适应不同阅览室布局、不同高校图书馆的座位种类、座位的不同使用状态。

3.3 扫码入座模块设计

学生通过小程序展示的图书馆座位使用情况状态图示，寻找到可以使用的座位，系统先进行学生身份的验证，验证扫码的用户是否属于本校的学生或者是否是具有本图书馆学习阅览权限的用户。验证通过后系统再对扫码的用户的进行位置验证，系统使用微信小程序提供的位置验证接口获取将要使用座位用户的位置信息，与后台预留的图书馆坐标进行距离测算。如果测算距离小于被允许的范围，系统将允许小程序调起扫码组件，继续扫码验证当前需要入座的座位信息。验证通过后，学生可以选择“学习 30 分钟”、“学习 60 分钟”、“学习 120 分钟”等，学生获取到座位一定时长的使用权限。至此，扫码入座流程结束。

3.4 数据库设计

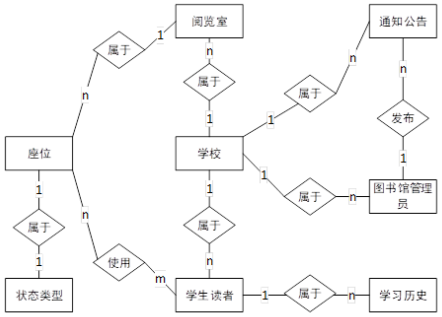


图 2 数据库实体-联系（E-R）图

学校、学生读者、学习历史、通知公告、阅览室、图书馆管理员、座位、状态类型八个实体之间的关系如图 2 所示。

4 系统实现关键技术

4.1 系统前后端技术

系统采用面向对象的方法开发整个软件系统，系统主要分为微信小程序端、网页管理端、后台三个部分。小程序前端开发采用微信开发者工具，运用 JavaScript 开发语言；网页管理端主要使用的技术有 HTML、CSS、JavaScript 等开发技术；系统后台主要采用的技术有 Spring Boot 框架、MyBatis、Redis 缓存技术搭建后台。

4.2 座位布局算法

由于每个用户的微信小程序客户端的分辨率的不同，如何解决在不同屏幕分辨率下友好的展示阅览室座位布局是小程序端需要解决的问题。算法步骤如下：

步骤 1　循环遍历座位列表数据，比较每个座位 X 坐标，得出最大轴 X 轴坐标；

步骤 2　循环遍历座位列表数据，比较每个座位 Y 坐标，得出最大轴 Y 轴坐标；

步骤 3　根据每个座位图标像素大小计算出未进行比例变换时全部座位在 X 轴方向所占宽度和 Y 轴方向所宽度；

步骤 4　步骤三得出的宽度与相应的微信小程序用户客户端屏幕分辨率大小进行比较，如果大于用户客户端屏幕分辨率，执行步骤五；

步骤 5　座位大小进行一定比例的缩小，使得不同阅览室座位布局中座位在不同分辨率屏幕完全友好的展示出来。

5 系统应用效果

5.1 阅览室管理页面

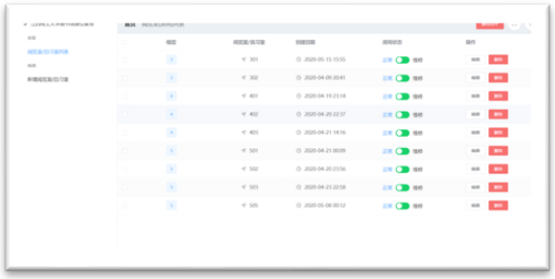


图 3 管理员阅览室管理页面

如图3所示阅览室(自习室)管理页面，页面主要展示楼层、阅览室号、阅览室创建时间、阅览室状态等信息列表，图书馆管理员可以点击“编辑”、“删除”按钮对阅览室进行操作。

5.2 座位管理页面

图书馆管理员可以在电脑端打开创建座位布局页面，如图 4 所示。在这个页面管理员用户可以根据图书馆的座位实际摆放情况，自定义创建座位，座位的状态也可根据实际情况选择普通座位、维修座位和过道座位状态。在小程序端，图书馆管理员可以查看阅览室内座位使用情况，如图 5 所示。

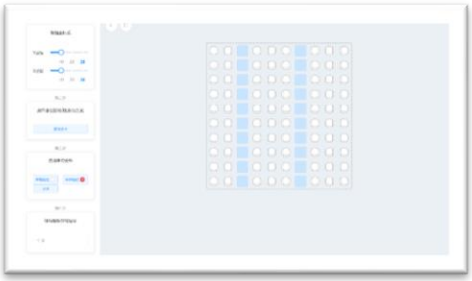


图 4 管理员修改阅览室信息页面

图 5 座位当前使用学生信息页面

5.3 扫码入座页面

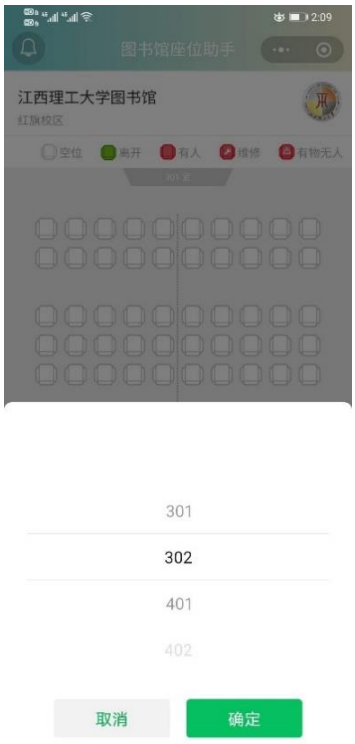


图 6 选择阅览室页面

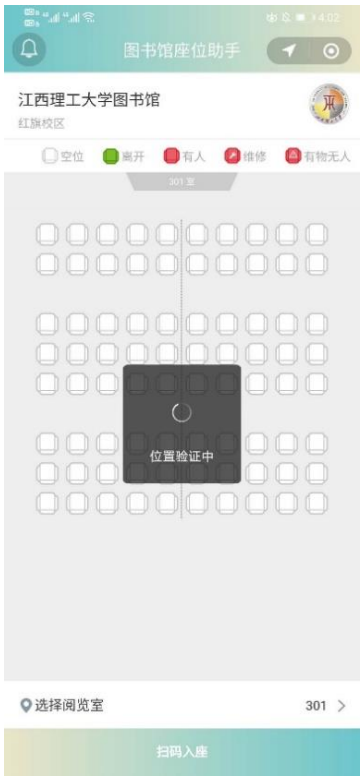


图 7 位置验证页面

扫码入座主要分为三步，学生用户首先点击首页下方选择器组件选择阅览室（如图 6 所示），然后进行位置验证（如图 7 所示），页面提示“位置中”提示框，位置信息验证通过后，调起微信小程序扫码能力页面，扫描图书馆内座位对应的座位二维码后，位置信息、个人信息、扫码座位信息验证全部通过后，页面弹出选择时间弹窗，有“学习 30 分钟”、“学习 60 分钟”、“学习 120 分钟”、“离开 15 分钟”、“离开 45 分钟”等按钮，选择学习的时间所示，扫码入座流程完成。

6 结语

本课题基于微信小程序的图书馆座位管理系统，针对各大高校面临的图书馆座位硬件资源不足的情况，提出软件系统管理解决方案。系统采用微信小程序客户端和网页的形式向学生和图书馆管理员提供图书馆座位的实时状态。系统在一定程度上讲，降低了高校图书馆座位资源的浪费，提升了现有的座位的使用效率。提高了图书馆在一定时间段的容纳使用座位学习的学生总数量。因为时间、服务器硬件资源等原因的限制，基于微信的图书馆座位管理系统还存在可以改进和优化的空间。

参考文献

[1] 国家统计局中国. 2019中国统计年鉴[M]. 中国统计出版社, 2019.

[2] 何宇蝶. 当代大学生考研率上升的原因[J]. 管理观察, 2019(34):137-138.

[3] 陆有丽, 邓凯航, 李雯婧, 等. 基于微信小程序的图书馆座位预约系统的设计与实现[J]. 湖南理工学院学报(自然科学版), 2020,33(01):29-33.

[4] 尹相权. 基于图书馆座位管理系统的用户行为分析[J]. 数字图书馆论坛, 2019(10):37-44.

[5] 蒋谢芳, 马璇, 王长浩, 等. 智慧图书馆座位管理系统设计与实现[J]. 数字技术与应用, 2019,37(06):158-159.

[6] 张毅. 基于微信小程序的图书馆座位管理系统[J]. 新世纪图书馆, 2019(第8期):62-65.

[7] 董桂娟. 浅谈高校图书馆座位资源紧缺现象及对策[J]. 吉林化工学院学报, 2011(第8期):66-67.