Computer Science and Application 计算机科学与应用, 2025, 15(6), 178-189
Published Online June 2025 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2025.156168

Hans汉斯

基于微信小程序停车场车位预约系统的设计与实现

汪俊龙，毛逸苇，阮天乐，周诗源*，邓　伟

嘉兴南湖学院信息工程学院，浙江　嘉兴

收稿日期：2025年5月22日；录用日期：2025年6月20日；发布日期：2025年6月27日

摘　要

如今私家车出行已经成为大多数人出行的首选方式，但是车位的难以寻找与不文明停车行为的问题也在日益严重。针对以上问题，本文开发了一款基于微信小程序的停车场车位预约与管理一体化系统，该系统对车位采用预约模式，用户可以通过系统查看车位信息并预约车位，提前锁定车位，并且可以自行取消或续约，在系统预约端实时更新，便于用户查看车位信息。同时，对约而不来等违规行为的用户进行限制性预约处理，避免车位资源的浪费。

关键词

停车场，车位管理，预约系统，微信小程序

Design and Implementation of a Parking Space Reservation System Based on WeChat Mini Programs

Junlong Wang, Yiwei Mao, Tianle Ruan, Shiyuan Zhou*, Wei Deng

College of Information Engineering, Jiaxing Nanhu University, Jiaxing Zhejiang

Received: May 22nd, 2025; accepted: Jun. 20th, 2025; published: Jun. 27th, 2025

Abstract

With private cars becoming the preferred mode of transportation for most people, the difficulty in finding parking spaces and the prevalence of uncivilized parking behaviors have become increasingly

*通讯作者。

文章引用: 汪俊龙, 毛逸苇, 阮天乐, 周诗源, 邓伟. 基于微信小程序停车场车位预约系统的设计与实现[J]. 计算机科学与应用, 2025, 15(6): 178-189. DOI: 10.12677/csa.2025.156168

severe. To address these issues, this paper develops an integrated parking space reservation and management system based on Mini program. The system adopts a reservation-based model, allowing users to check parking space availability, book slots in advance, and cancel or extend reservations as needed. Real-time updates on the reservation platform ensure users can access the latest parking information. Additionally, the system imposes restrictions on users who violate rules (e.g., no-shows after booking) to prevent the waste of parking resources.

Keywords

Parking Lot, Parking Space Management, Reservation System, WeChat Mini Program

1. 引言

随着我国居民生活水平的上升，汽车已经成为大多数人出行的重要方式之一，据国家统计局数据显示，2023 年，居民每百户家用汽车拥有量为 49.7 辆[1]，但与之形成鲜明对比的是，我国多数省会城市的停车缺口约达到 50%，而现有停车资源的利用率却不足 50% [2]。当前泊位资源配置存在三大结构性矛盾[3]：一是空间分布的碎片化特征与出行需求的集聚化趋势产生错配；二是传统人工管理模式与智慧交通发展要求存在代际差距；三是潮汐式停车需求波动与固定供给模式形成效能损耗。尤为突出的是，停车场缺乏数字化预约系统和动态调配机制，这种供需结构上的不匹配，使得固定供给的车位在高峰时段供不应求，而在低谷时段则大量闲置，造成资源浪费[4]。构建智能感知、动态调配、精准预约的智慧停车生态系统，已成为提升城市治理效能、解决泊位资源配置结构问题、优化出行服务体验的关键突破口。

2. 传统停车场车位管理的弊端

传统停车场的车位管理依赖于人工指引，人工管理，这样的管理方式存在以下问题：

2.1. 车位资源的浪费

随着我国人均 GDP 的增长，越来越多的家庭拥有私家车且有开车出行通勤的习惯。这对城市停车场车位的需求大大增加。然而许多停车场在驾驶员停车时，依靠停车场工作人员干预，告知驾驶员何处有车位何处无车位，但却无法做的及时性的车位资源更新，有时车辆已经驶离，有空闲车位的情况下，工作人员无法及时得知，容易造成车位资源的浪费。当停车场车位紧缺时，也容易出现争抢车位而导致的驾驶员之间的矛盾。

2.2. 难以应对高峰时期的车位需求

在节假日期间，出行人数会大大增加，对车位的需求也会大大增加。传统的管理方式面对节假日时大幅增长的出行车辆，容易造成停车场内拥堵，部分车位未能及时利用，使得驾驶员花费大量时间寻找车位。

2.3. 无法监控车位的使用状态

在传统的停车场管理系统下，停车场管理人员无法实时监测车位的使用情况，停车场内可能存在不

文明停车行为，例如部分车主将车停在两个车位之间，占用两个车位；部分人群存在恶意占车位；还有将非机动车停靠在机动车位里等行为。这导致了许多本可以使用的车位被浪费，大大影响了居民的出行体验以及停车场的管理难度。

2.4. 缺少数据分析支持

在如今的停车场车位管理问题中，缺乏数据分析支持是一个较为重要的问题。停车场管理员无法通过数据了解附近街区居民的用车停车习惯，在传统管理方式下会面临许多管理问题，影响了停车场车位管理优化。

3. 车位预约系统的可行性分析

3.1. WeChat Mini Program 框架选择

Mini program 是微信小程序官方提供的开发框架，是一个通用的网络通信框架，采用 IT 界推崇的 MVVM 模式[5]，框架核心是一个实时响应的数据绑定系统。此系统分为视图层(View)和逻辑层(APP Service)，此框架的作用就是保障视图层视图和逻辑层的数据同步。当需要数据更新时，只须在逻辑层修改数据，视图层就会自动响应并更新视图。它支持使用 JavaScript、WXML 和 WXSS 等技术进行开发。Mini program 框架还提供了丰富的 API 和组件，方便开发者快速构建小程序页面和功能，以及微信官方提供了详细的开发文档，能够帮助我们在服务的搭建、管理上有较好的可靠性，能够轻松处理用户登录，车位预约逻辑等核心功能。

3.2. 设计基础架构

微信小程序利用传统的 C/S 网络架构为基础，完成客户端和服务端直接相连。这种点对点的连接方式最为突出的特点就是数据传输安全高效。微信小程序网络架构为小程序与服务器之间的数据交互提供了速度和安全的保障。微信小程序网络架构在 C/S 架构的基础上利用动态语言和算法突破客户端必须安装客户端服务软件的限制。微信小程序采用 MINA [6]框架的实时数据绑定和算法完成加载和运行同步，类似于 DCloud 流应用。当需要发布最新版的微信小程序时，只需要把完善好的微信小程序传送到服务器供用户调用，正在使用该版本的用户不影响其继续使用，因为代码包已经缓存到本地[7]。

3.3. 前端框架选择

小程序在前端(逻辑层)中主要使用微信所提供的基于 WXML、WXSS 和 JavaScript 实现的框架[8]，可实现快速开发界面和逻辑，其便捷易学的特性能够使我们前端的开发更加高效。通过 Mini program 框架，我们能够通过改变逻辑层的代码，来使视图层自动更新响应视图，快速搭建直观的用户界面，提供车位实时使用情况、预约功能，不文明停车行为举报功能等，为驾驶员提供便捷的预约停车服务。

3.4. 后端框架选择

在后端方面，小程序使用云开发平台作为服务提供商，来实现对移动端提供数据存储和处理等基本功能。微信云开发提供了丰富的 API，且经过了广泛的实际应用和时间的考验，为我们的小程序提供了一个可靠的数据储存方案，且微信云平台能够迅速相应用户的查询请求以及举报请求，能够帮助我们快速更新车位的使用状态，以及快速纠正停车场内的车位使用问题。

4. 系统功能设计

整个系统平台分为用户平台与管理者平台两个模块，具体功能如下：

4.1. 用户平台

用户平台是提供给各位驾驶员用于进行各项需求满足的交互平台。可以通过微信小程序搜索或者扫描二维码访问小程序，将用户的手机号作为身份识别标志注册用户平台账号并登入小程序，享受小程序所提供的以下功能：

(1) 车位使用情况的查看以及预约。驾驶员登入系统后，可以通过小程序查看停车场内车位分布的地图，以及各区车位的预约使用情况。驾驶员可根据自身出行需求来点击车位进行车位预约，被预约的车位将锁定，不可再被其它用户预约。

(2) 已预约车位的取消预约。若驾驶员出行计划变更，用户可在预约时间开始前至预约时间开始后的前 15 分钟内，通过小程序取消预约。

(3) 举报违规停车。若用户在停车过程中发现有车辆未停入预约车位，随意占用他人车位，未停入指定区域等不文明停车行为，可通过举报系统进行反馈，系统将酌情扣除违停车主的信用分，使其短期内无法进行预约。

(4) 用户辅助功能。例如我的预约，我的资料，意见反馈等。用户可以通过我的预约查看车位的历史预约记录；可以通过我的资料查看个人信息，修改个人信息；可以通过意见反馈传达对于停车场的整改意见。

(5) 地图导航功能。在检测到用户到达停车场后，可根据实时定位数据，通过导航按钮触发，自动生成最优行驶路线。

4.2. 管理者平台

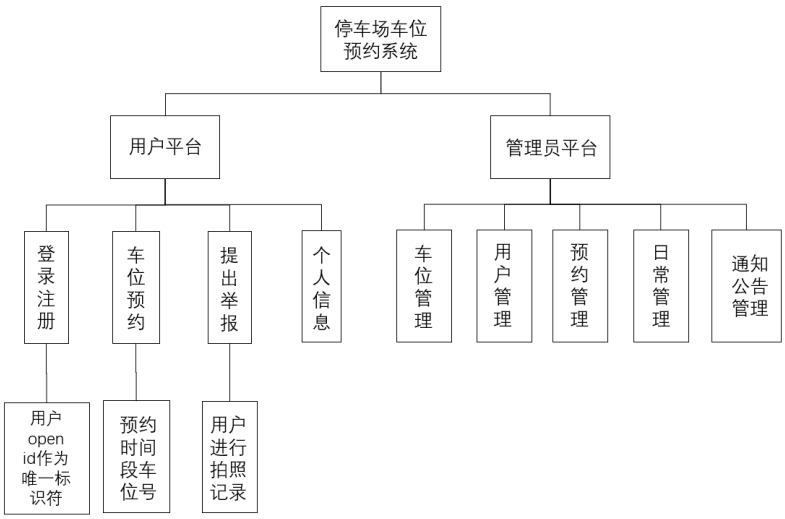

Figure 1. Functional design diagram
图 1. 功能设计图

管理者平台只能由管理人员登录，用于管理系统信息，包括以下功能：

(1) 车位管理。在车位管理中，管理人员可以对车位的信息进行修改。

(2) 用户及其预约管理。管理人员可以查看所有用户的车位预约信息，并以管理员的身份进行预约信息的修改。

(3) 日常管理。管理人员可通过举报信息核实不文明停车行为，对违停用户或举报不实信息用户进行扣除信用分的警告，若信用分过低，将被拉入黑名单，短时间内无法进行预约。

(4) 通知公告。管理人员可通过通知公告发布停车场公告，向用户传达停车场的日常通知以及出发管理信息。功能设计如图 1 所示。

5. 小程序的实现

5.1. 小程序的架构

整个小程序的框架系统分为两部分：逻辑层(App Service)和视图层(View)。小程序提供了自己的视图层描述语言 WXML 和 WXSS，以及基于 JavaScript 的逻辑层框架，并在视图层与逻辑层间提供了数据传输和事件系统。小程序架构如图 2 所示。

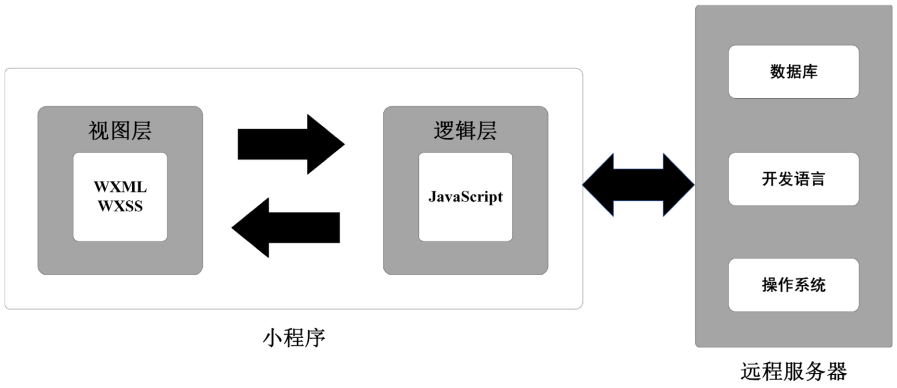

Figure 2. Mini program architecture diagram
图 2. 小程序架构图

小程序框架中的逻辑层使用 JavaScript 引擎为小程序提供开发 JavaScript 代码的运行环境以及微信小程序的特有功能。逻辑层将数据进行处理后发送给视图层，同时接受视图层的事件反馈。

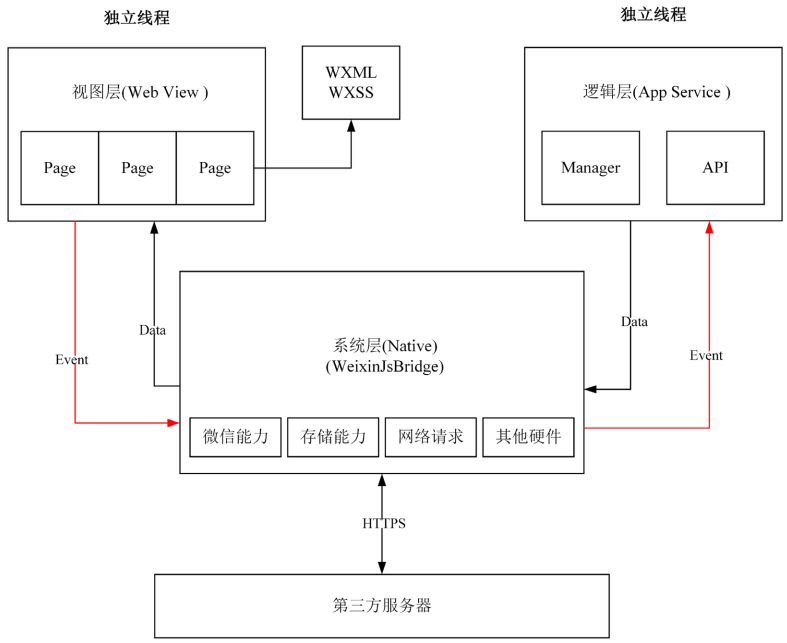

Figure 3. Diagram of the working principle of the Mini program
图 3. 小程序工作原理图

小程序框架中的视图层框架的视图层由 WXML 与 WXSS 编写，由组件来进行展示。将逻辑层的数据反映成视图，同时将视图层的事件发送给逻辑层。WXML (WeiXin Markup language)用于描述页面的结构。WXS (WeiXin Script)是小程序的一套脚本语言，结合 WXML，可以构建出页面的结构。WXSS (WeiXin Style Sheet)用于描述页面的样式。小程序工作原理图，如图 3 所示。

5.2. 小程序预约功能实现界面

如图 4 所示，在小程序预约界面中，用户可实时查看车位预约状态。具体操作流程为：首先选择目标车位，系统将自动跳转至信息确认页面；其次，用户需选定预约日期和时间段(系统将自动调用用户预设的姓名及车牌号信息)；最后点击“确认预约”按钮完成操作。预约成功后，对应车位状态将实时更新为“已预约”标识。

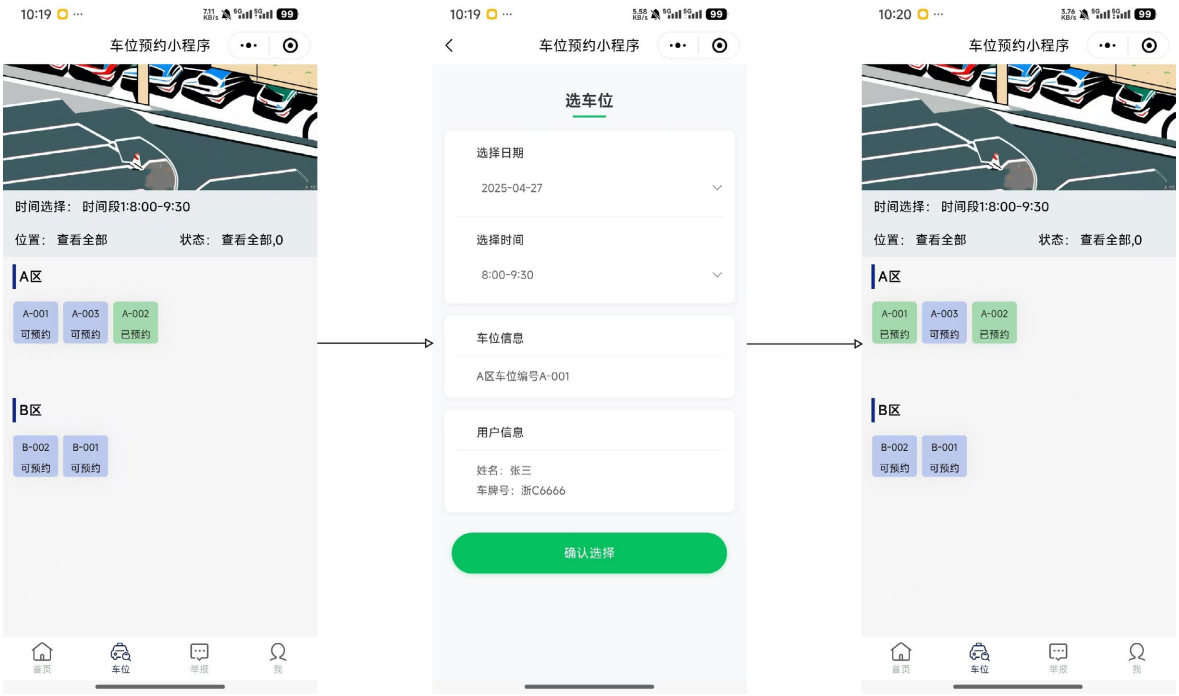

Figure 4. Appointment fulfillment interface diagram
图 4. 预约实现界面图

车位预约部分关键代码及其注释：

```
// 定义一个名为 showxq 的函数，接收事件对象 e 作为参数
showxq(e) {
    // 打印事件当前目标元素的 id (通常是触发事件的元素的 id)
    console.log(e.currentTarget.id)
    // 从数据库集合"tsgzw"中获取指定 id 的文档(e.currentTarget.id 是文档 id)
    db.collection("tsgzw").doc(e.currentTarget.id).get().then(res => {
        // 将获取到的文档数据中的 zwzt 字段更新为 sj 数组中特定索引位置的第一个元素
        // this.data.sjxxindex 是当前组件/页面数据中的索引值
        res.data.zwzt = res.data.sj[this.data.sjxxindex][0]
        // 打印更新后的 zwzt 值
```

```
            console.log(res.data.zwzt)
            // 判断 zwzt 的值是否为 0
            if (res.data.zwzt == 0) {
                // 如果 zwzt 为 0，跳转到 zwyd 页面，并携带两个参数：
                // list_id: 当前目标元素的 id
                // sjid: 当前组件/页面数据中的索引值
                wx.navigateTo({
                    url: '../zwyd/zwyd?list_id=' + e.currentTarget.id + "&sjid=" + this.data.sjxxindex,
                })
            } else {
                // 如果 zwzt 不为 0，显示一个错误提示，内容为"不可预约"，持续 2 秒
                wx.showToast({
                    title: '不可预约',
                    icon: 'error',
                    duration: 2000
                })
            }
        })
```

}, 通过当用户点击某个座位时，检查该座位在当前选择的时间段是否可以预约如果状态是 0 (可预约)：跳转到预约页面，并带上座位编号和时间段信息。如果状态不是 0 (不可预约)：弹出一个提示框，告诉用户“不可预约”。

5.3. 小程序举报功能实现界面

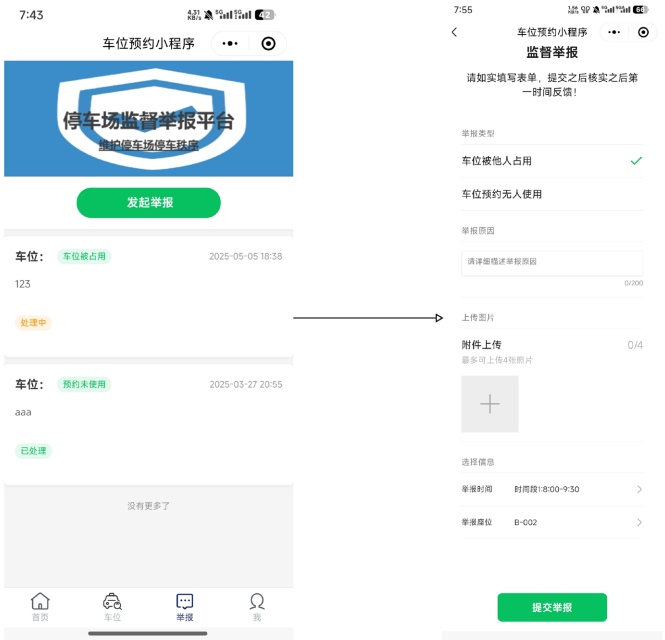

Figure 5. Diagram of the supervision and reporting interface
图 5. 监督举报界面图

如图 5 所示，在举报功能模块中，用户可通过点击“发起举报”按钮进入监督举报页面。该页面支持用户针对平台其他用户的不同违规行为进行举报操作。当用户完成举报表单填写并提交后，系统后台将自动接收并处理相关举报信息。此外，系统还为用户提供了举报记录查询功能，用户可随时返回举报页面查看历史举报记录。

举报功能关键代码及其注释：

```
submitForm() {
    // 表单验证不通过则直接返回
    if (!this.validateForm()) return
    // 设置提交状态为 true(防止重复提交)
    this.setData({ submitting: true })
    // 构建举报数据对象
    const reportData = {
      openid: this.data.openid,   // 用户唯一标识
      yy: this.data.jbyy,             // 举报原因
      jbzw: this.data.items[this.data.jbzw].value,   // 举报座位信息
      zp: this.data.files,            // 上传的凭证图片
      jblx: this.data.formData.radio,   // 举报类型(单选值)
      sjd: this.data.sjxxIndex,   // 时间段索引
      cljg: '',                         // 初始处理结果为空
      _createTime: db.serverDate(),   // 使用服务器时间戳
      status: 'pending'                 // 初始状态为待处理
    }
    // 数据库操作
    db.collection('jdjb').add({   // 向 jdjb 集合添加数据
      data: reportData
    }).then(() => {
      // 提交成功提示
      wx.showToast({
        title: '举报成功',
        icon: 'success',
        duration: 2000
      })
      // 2 秒后跳转首页
      setTimeout(() => {
        wx.switchTab({
          url: '../index/index'
        })
      }, 2000)
    }).catch(err => {
      // 错误处理
```

```
            console.error('提交举报失败：', err)
            wx.showToast({
              title: '提交失败，请重试',
              icon: 'none'
            })
          }).finally(() => {
            // 无论成功失败都重置提交状态
            this.setData({ submitting: false })
          })
      }
```

5.4. 小程序后台管理界面

Figure 6. Diagram of the backend management interface
图 6. 后台管理界面图

如图 6 所示，在小程序后台管理界面，顶部导航栏包含两个选项卡：“车位管理”和“举报管理”，通过点击相应选项卡可在两个管理模块之间进行切换。处于选中状态的选项卡会通过样式变化进行区分，

方便管理员识别当前所在的管理模块，并对对应模块进行操作。

系统首先调用云函数 login 获取用户的微信 openid (微信用户唯一标识)，以此验证用户的管理员身份。在身份验证通过后，系统将从数据库中导出全部车位信息及相关举报记录，并将这些数据动态呈现在后台管理界面中。

在界面设计方面，系统采用顶部导航栏与内容区域相结合的经典布局模式。信息展示采用卡片式设计，每个车位信息及举报记录均以独立卡片的形式呈现，确保信息的清晰可辨。管理员在界面上的所有操作(如数据修改、状态更新等)都将实时同步至数据库，保证数据的一致性。

针对举报信息处理功能，系统会在数据库相应记录中新增“处理结果”字段，用于记录管理员对举报事项的处理意见及处理状态。这一设计实现了举报信息的闭环管理，为后续查询和统计提供了数据支持。

后台管理界面关键代码及其注释：

```
// 检查当前用户是否是管理员
checkAdmin: function() {
    // 显示加载状态
    this.setData({ loading: true });

    // 调用云函数获取用户的 openid
    wx.cloud.callFunction({
      name: 'login',   // 调用的云函数名称(通常是获取用户 openid 的云函数)
      success: res => {
        console.log('获取 openid 成功：', res);
        const openid = res.result.openid;   // 从返回结果中提取 openid
        this.setData({ openid });   // 将 openid 存入页面/组件的 data 中
        // 查询数据库的'admin'集合，检查当前用户是否是管理员
        db.collection('admin')
          .where({ openid: openid })   // 查询条件：openid 匹配当前用户
          .get()   // 执行查询
          .then(res => {
            // 判断是否是管理员：如果查询结果有数据(res.data.length > 0)，则是管理员
            const isAdmin = res.data && res.data.length > 0;
            this.setData({ isAdmin });   // 更新 isAdmin 状态
            if (isAdmin) {
              // 如果是管理员，加载初始数据(如待处理的举报列表等)
              this.loadData();
            }
            this.setData({ loading: false });
          })
          .catch(err => {
            console.error('检查管理员权限失败：', err);
            this.setData({ loading: false });   // 关闭加载状态
```

```
                wx.showToast({
                  title: '检查权限失败',  // 提示用户权限检查失败
                  icon: 'none'  // 不显示图标
                });
              });
          },
          fail: err => {
            console.error('获取 openid 失败：', err);
            this.setData({ loading: false });
            wx.showToast({
              title: '获取用户信息失败',
              icon: 'none'
            });
          }
        });
    },
```

6. 存在的问题与对策

6.1. 用户举报的监督机制需要进一步改善

单纯依靠用户举报的监督机制(如车位违规占用及预约未使用等情形上报)存在固有缺陷：其一，举报行为具有非强制性特征，导致监管覆盖率不足；其二，信息反馈存在时间滞后性，难以实现实时管控。这种事后追责模式无法从根本上预防乱停乱放行为的发生。可以引入自动化车位监测技术，通过布置摄像头和传感器等硬件监测，检测车辆是否违规停放，并与预约数据进行比对。并且对频繁违规用户建立信用黑名单，限制预约权限。

6.2. 用户体验需进一步优化

本系统的主要功能是停车场车位预约，并不能根据用户的偏好或历史记录为用户提供个性化服务。可以通过引入用户反馈投稿机制，以便系统及时了解用户的建议，以便系统进行改进。另外，车位预约功能设置了严格的预约时段，用户无法自定义预约时长，可以通过提供更灵活的预约时段选项，以此来优化用户体验。

7. 结语

本研究基于微信小程序平台设计并实现了一套智能化的车位预约管理系统，旨在解决传统停车场管理中存在的资源分配不均、使用效率低下等问题。系统采用微信官方提供的 Mini program 框架作为开发基础，充分利用其 MVVM 架构的优势，实现了视图层与逻辑层的高效数据同步。在前端开发中，我们运用 WXML、WXSS 和 JavaScript 技术栈，构建了直观友好的用户界面；后端则依托微信云开发平台，确保了数据存储和处理的可靠性。

本系统的主要创新点体现在三个方面：首先，通过 C/S 架构的点对点连接方式，实现了数据传输的安全高效；其次，采用动态数据绑定技术，确保了车位状态的实时更新；最后，引入信用评分机制和用户举报监督机制，有效规范了停车行为。

尽管当前系统在用户体验和性能优化方面仍有提升空间，但通过持续的技术迭代和用户反馈收集，我们相信可以不断完善系统功能。未来，我们将提高系统的扩展性和用户友好性，推动系统向更智能、更高效的方向发展。希望本研究成果能够为智慧城市建设中的交通管理问题提供有益参考。

基金项目

嘉兴南湖学院 2024 年国家级大学生创新创业训练计划项目(202413291010)，嘉兴市应用性基础研究项目(2023AY11026)，浙江省文化广电和旅游厅科研与创作项目(2024KYY029)，浙江省高等教育学会 2025 年度高等教育研究课题(KT2025186)，嘉兴南湖学院 2024 年教育教学改革研究项目(重点) (22042024107)，嘉兴南湖学院科研启动基金(QD61220029)。

参考文献

[1] 国家统计局. 新中国 75 年经济社会发展成就系列报告之五[EB/OL]. https://www.stats.gov.cn/sj/sjjd/202409/t20240911_1956379.html, 2024-09-11.

[2] 高照, 高丽燃, 史未名. 共享停车缓解居住区停车问题[C]//中国城市规划学会, 杭州市人民政府. 共享与品质——2018 中国城市规划年会论文集(06 城市交通规划). 2018: 731-739.

[3] 年佟, 习聪玲, 孟以媛, 等. 基于微信小程序的车位预约系统设计与实现[J]. 电脑编程技巧与维护, 2024(9): 68-70+99.

[4] 李盼道, 李洋. 配置效率驱动下停车资源的属性、产权及供给模式[J]. 西南大学学报(社会科学版), 2018, 20(2): 27-38.

[5] 游俊慧. MVC、MVP、MVVM 三种架构模式的对比[J]. 办公自动化, 2020, 25(22): 11-12+27.

[6] 蔡谊. 基于 MINA 框架的网络管理软件设计[J]. 通信技术, 2013, 46(3): 115-117.

[7] 李哲, 周灵. 微信小程序的架构与开发浅析[J]. 福建电脑, 2019, 35(12): 66-69.

[8] 何露莹. 大型智能停车场管理系统的设计与实现[D]: [硕士学位论文]. 桂林: 电子科技大学, 2023.