密级：公开

高校固定资产管理系统的设计与实现

Design and Implementation of University Fixed Assets Management System

学　　院：软件学院

学　　号：161203726

专业班级：软件工程 1607 班

学生姓名：刘靖诗

指导教师：邵中

2020年06月

摘 要

高校的固定资产是发展高等教育的物质基础，随着高校办学规模的逐渐扩大扩大必然会导致高校的固定资产的种类和数量不断增多，固定资产的管理也会变得越来越复杂，传统的固定资产管理模式已经无法方便快捷的完成对固定资产的管理，高校固定资产的管理已从传统模式转变为信息化管理模式。

本文以 PC 端和移动端相结合的方式进行固定资产的管理，实现了资产信息管理、资产盘点功能、审批管理功能以及部门管理、职位管理、人员管理、用户管理等基本功能，移动端首页可以发布帖子和回复帖子，进行用户之间交流。利用 GPS 定位以及二维码管理，为每个固定资产生成单独的二维码及其位置信息。将二维码的加密、资产图片的自动审核作为设计重点。采用 B/S 架构，以浏览器、微信小程序、手机 APP 作为客户端，以 Apache、Tornado 作为 Web 服务器，使用 MVC 设计模式以及 Vue 单页应用进行开发，提高了代码的可读性以及系统的可维护性，并申请 SSL 证书，使用 HTTPS 协议进行 Request 请求，WSS 协议进行 WebSocket 连接，保证了数据传输的安全性，采用 MySQL 数据库进行数据存储，并关闭了外网对数据库的访问，保证了数据的安全性，前端使用 Bootstrap+Vue.js+layui 作为主体框架，实现了响应式以及数据的双向绑定，使用 uni-app 进行移动端开发，可以一键生成手机 APP 以及微信小程序。

目前高校固定资产管理系统已经初步开发完成并部署使用，系统设计合理，在服务器上稳定运行，并且满足高校对于固定资产管理的需求，提高了高校固定资产的效率。

关键词：固定资产；管理系统；二维码；图片相似度；

Abstract

The fixed assets of colleges and universities are the material basis for the development of higher education,With the gradual expansion of the scale of running a school, the types and quantity of fixed assets of colleges and universities will inevitably increase, and the management of fixed assets will become more and more complex,The traditional fixed assets management mode has been unable to complete the management of fixed assets conveniently and quickly,The management of fixed assets in colleges and universities has changed from the traditional mode to the information management mode.

This paper combines PC terminal and mobile terminal to manage fixed assets,It has realized the functions of asset information management, asset inventory, examination and approval management, department management, position management, personnel management, user management and other basic functions,The mobile homepage can publish and reply posts for communication between users.GPS positioning and QR code management are used to generate a separate QR code and its location information for each fixed asset.Take the encryption of QR code and automatic audit of asset pictures as the key points of design.B/S architecture is adopted, with browser, WeChat applet and mobile APP as clients, Apache and Tornado as Web servers,Using MVC design pattern and Vue single page application to develop, improve the readability of code and system maintainability,And apply for SSL certificate, use HTTPS protocol for Request, WSS protocol for WebSocket connection, to ensure the security of data transmission,MySQL database is used for data storage, and external network access to the database is closed to ensure the security of the data,The front uses Bootstrap+Vue.js+Layui as the main framework to realize the two-way binding of response and data,Using uni-app for mobile development, you can generate mobile app and WeChat applet with one click.

At present, the fixed assets management system in colleges and universities has been preliminarily developed and deployed,The system design is reasonable and runs stably on the server,It also meets the demand of universities for fixed assets management and improves the efficiency of fixed assets management.

Keywords: Fixed assets; Management system; QR code; Image similarity;

目　录

第1章 引 言

1.1 研究背景与意义

近几年随着我国经济的突飞猛进，高等教育事业得到了蓬勃发展。高校固定资产规模随之扩大，资产构成日趋复杂，导致管理难度也越来越大。加强高校固定资产管理不仅是为了确保国有资产的安全完整，也是高校自身发展的内在需要。因此，进一步提高高校固定资产管理水平和能力显得至关重要。

当前信息化技术已经被逐步引入到高校的固定资产管理工作中，在台账建档、资产调拨、折损计算等业务工作中取得了明显的效益，大幅度地提高了工作效率和质量。但是多数现有系统所提供的业务支持不够全面，偏重于资产登记、调拨、使用和折损处置等环节，而在清查、盘点、实物核对这一类需要大量人员参与的业务工作中却缺乏有效的工作模式和功能支撑，导致上述工作依然需要通过少数人员的繁重劳动方可完成，未能从根本上解决高校固定资产管理所面临的痛点问题。尽管目前以 RFID 智能标签结合移动射频采集设备的解决方案可以在一定程度上缓解资产清查盘点工作的困境，但相对高昂的应用成本则成为其应用推广所面临的主要障碍。

本课题以图形二维码和智能手机取代 RFID 标签及专用射频采集设备，给出一种低成本且便于广泛参与的高校固定资产管理解决方案及配套软件系统。课题的研究及实践进一步提升固定资产管理水平和质量，提高资产清查盘点的准确性，降低应用成本和工作人员的劳动强度，具有明确的实用价值和现实意义。

1.2 国内外研究现状

分别从同类产品的应用现状和相关技术研究进展两个方面对与课题相关的现存工作进行调研分析，分类归纳如下。

1.2.1 应用现状

固定资产管理信息系统是以固定资产为管理对象，以资产台帐为基础，以统筹使用，防止流失，加强管理监督并提升使用效率为目的的管理系信息系统。当前市场上存在大量的同类产品，本文重点对其中与移动端应用功能支持有关的 3 款产品进行分析。

“易点固定资产管理系统”由易点易动公司开发，主要功能分为资产管理、耗材管理、财务管理以及审批管理四大模块。该系统支持移动端 APP 接入，可以使用手机扫码完成资产盘点，但无法对固定资产地理位置进行 GPS 定位；后台可提供对 RFID 标签的功能支持，但需要单独购买相应设备。

“联想百应资产管理系统”是由联想公司开发的一款同类应用系统。功能分为资产管理和耗材管理两大模块，无法使用手机扫码进行资产盘点，但可以申请工作人员代盘点，需要支付代盘点费用，也可以直接使用联想商城购买耗材。

“简普固定资产管理软件”是一款由 ASP/C#开发的固定资产管理软件，其功能分为个人办公、资产管理和耗材管理三大模块，无法使用手机扫码进行资产盘点，界面简单，但支持支持盘点任务的短信通知。

1.2.2 研究现状

大部分固定资产管理相关系统采用 PC 端 B/S 架构模式以及 RFID 自动识别技术，在 2013 年以前，多数为 Java 开发的 C/S 架构模式，目前，已有小部分系统采用了移动端的形式。

在 2019 年张慕然[1]基于 ASP.NET 平台和 SQLServer 数据库实现浏览器与服务器(Browser/Server，B/S)架构的数据库系统，采用 Knockout 作为 JavaScript 库，EasyUI 作为 UI 库，采用面向对象的设计思想建立资产管理系统的数据模型、功能模型和分层模型和规范的业务管理流程。切实做到资产管理落实到人、落实到点、物物可查、事事能循。

姜宇飞[2]基于 Java 平台采用了 Spring MVC 开发模式，以 Apache Tomcat 为服务器，采用 MySQL 关联数据库，并通过 MyBatis 技术实现数据库的链接，实现了设备资产从采购到使用到维护到报废整个生命周期的一体化管理，使得设备资产的各个管理环节完美衔接，形成了更加科学合理的管理流程。

张华[3]在 2018 年利用非对称加密算法对 QR 二维码加密，生成了基于 DES 加密的 QR 二维码，并将私钥放置于服务端，公钥放置于客户端，在用户扫描二维码时进行解密，有效杜绝二维码的篡改、伪造等系列问题，提高里面二维码的安全性。

高硕[4]在 2019 年在其开发的基于内容的图像检索系统中，根据提取的主要特征，采用欧氏距离、直方图相交等方法对图像相似性进行度量，可方便地替换为马氏距离、棋盘格距离、切比雪夫距离、直方图相交法等多种度量方法，计算被检索图像与当前图像的相似度。

1.3 研究内容及主要工作

课题的主要目标是在 PC 端与移动端（手机 APP、微信小程序）实现一套完整的高校固定资产管理系统，可以投入实际运营。对于移动端而言，实现资产信息查看、资产入库、资产盘点、资产的借用与归还，在资产盘点的过程中，对于用户拍摄的照片进行裁剪，并与资产入库的图片进行相似度对比，防止用户胡乱拍摄。并提供了小型论坛交流场景，方便询问一些关于资产使用方面的注意事项，管理员也可以利用该场景发布一些通知。

系统 PC 端主要采用 ThinkPHP5 作为开发框架，使用 Bootstrap4 配合 Vue.js 以及 layui 组件完成 PC 端设计，数据处理采用 PHP 脚本语言，外加阿里云提供的短信接口 API 以及地图接口 API，完成了高校固定资产管理系统的 PC 端。

1.4 实验环境与条件

系统移动端使用微信开发平台以及 HBuilder X 配合生产原生 APP 以及微信小程序的发布，开发语言采用 JavaScript 和 Python；系统 PC 端使用 PHP 语言开发。服务端实验环境服务器采用 CentOS7.4，客户端环境采用 Windows10 Google Chrome、iOS10、Android9。

第 2 章　相关理论与技术分析

系统 PC 端基于 ThinkPHP5 框架，采用 MVC 设计模式，使用 Apache 作为 Web 服务器软件，使用 Bootstrap+Vue.js+layui+jQuery+ECharts 进行前端页面的开发，系统移动端基于 Tornado 作为服务端开发框架，uni-app 作为客户端开发框架，数据库采用 MySQL，PC 端使用 ThinkPHP 自带的 Db 类进行数据库操作，移动端使用本人基于 pymysql 封装的 Ljsmysql 类进行数据库操作，下面将对系统的主要技术进行介绍。

2.1 主要理论分析

2.1.1 加密算法

目前比较流行的加密算法有两种，分别是 AES 加密算法和 RSA 加密算法，两者的区别在于 AES 为对称加密算法，RSA 为非对称加密算法。

AES 算法是一种新型加密方法，是美国联邦政府采用的一种区块加密标准，具有更加可靠的加密过程和更加适合的密钥长度[5]，AES 加密和解密需要同一种秘钥，一般为固定长度的字符串，密钥长度可以是 128，192 或 256 位，加密过程是在一个 4×4 的字节矩阵上运作，加密时，各轮 AES 加密循环均包含 4 个步骤：AddRoundKey（矩阵中的每一个字节都与该次回合金钥做 XOR 运算）、SubBytes（通过一个非线性的替换函数，用查找表的方式把每个字节替换成对应的字节）、ShiftRows（将矩阵中的每个横列进行循环式移位）、MixColumns（为了充分混合矩阵中各个直行的操作）。

RSA 算法是一种非对称加密算法，是 1977 年由罗纳德·李维斯特（Ron Rivest）、阿迪·萨莫尔（Adi Shamir）和伦纳德·阿德曼（Leonard Adleman）一起提出的。RSA 的秘钥有两种分别是公钥和私钥，使用一种秘钥加密则必须使用另一种秘钥进行解密，秘钥长度一般为 1024、2048、3072、7680 或 15360 位，长度越长，安全性越高，世界上还没有任何可靠的攻击 RSA 算法的方式，只要其钥匙的长度足够长，用 RSA 加密的信息实际上是不能被解破的。

2.1.2 图片相似度计算

目前比较流行的图片相似度计算方法三种，分别是结构相似性度量、余弦相似度以及基于直方图计算相似度。

结构相似性度量是一种全参考的图像质量评价指标，分别从亮度、对比度、结构三个方面度量图像相似性，取值范围是[0, 1]，值越大，表示图像失真越小，比较方法是利用滑动窗将图像分块，采用高斯加权计算每一窗口的均值、方差以及协方差，然后计算对应块的结构相似度，最后将平均值作为两图像的结构相似性度量。

余弦相似度把图片表示成一个向量，通过计算向量之间的余弦距离来表征两张图片的相似度。该方法运算量较大，速度比结构相似性度量慢，但结果比其可靠。

颜色直方图是在许多图像检索系统中已获得广泛采用的颜色特征，算法简单方便[6]。单一通道的直方图相似值是通过获取颜色直方图数据，再使用巴氏系数算法计算出相似程度。

巴氏系数的算法如式(2-1)所示。

$$\rho(p,\ p') = \sum_{i=1}^{N} \sqrt{p(i)p'(i)} \tag{2-1}$$

式中p和p'分别表示原始与待比较的图像直方图数据，N表示直方图数据的个数，i表示当前直方图数据的索引，最终得到的ρ即是两图片的相似度。

2.2 主要前端技术

2.2.1 WebSocket 通讯技术

Websocket 是 HTML5 新标准中的通信机制，能够实现稳定全双工实时通信，具有简洁高效的特点[7]。提供了 WebSocket 接口的相关定义，允许服务端主动向客户端推送数据，不必使用 Ajax 轮询来获取服务端主动发送的数据。在 WebSocket API 中，浏览器和服务器只需要要做一个握手的动作，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。本文的移动端和服务端就是使用了 WebSocket 通讯技术。

2.2.2 Vue.js 框架

Vue.js 是一个非常典型的 MVVM 框架，模型只是一个 JavaScript 对象，如果修改了模型视图会自动更新，这种设计让状态的管理变得简单而且直观[8]。使用 Object.defineProperty，通过设定对象属性的 getter 和 setter 方法来监听相应的数据变化，通过 getter 进行依赖收集，每个 setter 方法都是一个观察者，在数据发生变

化时触发相应的监听回调，通知订阅者进行视图更新。这样就实现了数据的双向绑定。本文主要在表单上使用 Vue.js 实现数据的双向绑定。

2.2.3 uni-app

uni-app 是基于 Vue.js 框架所开发的跨平台应用前端[9]。开发者编写一套代码，可发布到 iOS、Android、H5、以及各种小程序、快应用等多个平台。uni-app 是目前跨端成熟度和案例数量最多的框架。每一个页面对应一个.vue 文件，里面分为 HTML、CSS 和 JavaScript 三部分，HTML 和 CSS 控制着该页面的内容与样式，JavaScript 部分包含诸多生命周期函数，监听 APP 从打开到关闭的每一步动作。本文的移动端则使用 uni-app 进行开发。

2.2.4 ljspopup

ljspopup 是本人使用 JavaScript 语言基于 jQuery 编写的弹出层组件，界面风格模仿了 Mac OS 的窗口，与 Mac OS 的窗口有 90%的相似度。调用时只需要一句 JavaScript 代码即可实现弹出层，具有关闭、最大化、最小化、移动等功能，可以无限层弹出，最小化后可以在右下角还原。

2.3 后端主要技术

2.3.1 Tornado 框架

Tornado 全称 Tornado Web Server，是一个用 Python 语言写成的 Web 服务器兼 Web 应用框架，由 FriendFeed 公司在自己的网站 FriendFeed 中使用，被 Facebook 收购以后框架以开源软件形式开放给大众[10]。

Tornado 采用异步非阻塞 IO 的处理方式，处理速度非常快，具有很强的抗负载能力。本文主要使用 Tornado 中的 WebSocket 创建 Socket 服务端，配合使用 Apache 做反向代理服务器部署。

2.3.2 OpenCV

OpenCV 是一个开源的可以跨平台的计算机视觉库，可以运行在多种操作系统中，并且为多种计算机语言提供了接口，可以实现计算机视觉、图像处理方面的很多算法[11]。本文的图片核对算法就是利用了 OpenCV 的 Python 接口来进行图片处理以及颜色直方图的数据处理

2.3.3 ThinkPHP5 框架

ThinkPHP5 是一个免费开源、快捷简单的轻量级 PHP 开发框架，基于 MVC 架构模式，高度封装了数据库的 CURD 操作和常用 API[12]。本文 PC 端则使用了 ThinkPHP5 进行开发，Model 层和 Controller 层用来编写后端代码，View 编写前端代码，由于使用了 ThinkPHP5 的 Route，所以在前端并未使用 Vue.js 的 vue-router。

2.3.4 Ljsmysql 数据库操作类

Ljsmysql 是本人使用 Python 语言基于 pymysql 编写的 MySQL 数据库操作类，使用风格模仿了 ThinkPHP5 中的 Db 数据库操作类，无需编写 SQL 语句，只需要使用 insert、delete、update、select、order、page 等函数即可实现最基本的增删改查以及排序和分页查询等操作。

第 3 章　系统需求分析与总体设计

在系统开发之前，必须对系统的需求以及总体设计进行必要的分析与规划，本章就会对系统需求分析、系统总体设计以及数据库设计进行详细的介绍。

3.1 系统需求分析

3.1.1 业务流程分析

高校固定资产管理基本的业务流程图如图 3-1 所示。

管理员用户需要先进行资产登记，也就是资产入库，登记资产的相关信息，系统会自动生成二维码，工作人员下载后打印贴到相应的固定资产上面，即可完成资产入库。

当所有的资产全部登记到系统之后，学生和老师会进行资产的借用/归还操作，使用手机扫描资产二维码或者使用电脑登陆 PC 端后台，选择借用或者归还，然后等待管理员审核通过。

对于管理员用户来说，可以直接将资产借用给某人，也可以选择审核通过某人的借用或归还的申请，资产的状态就会从审核中变成在用或闲置。等到了需要进行资产盘点的时候，管理员需要在后台创建盘点单，指定盘点人员。接到盘点任务的用户需要进行资产盘点，对盘点任务的资产进行扫码拍照，等待管理员审核，管理员需要对系统无法判断的资产进行核对，如果资产不合格资产需要用户重新进行盘点，等到所有资产全部盘点合格之后盘点任务完成。

对于系统管理用户来说，可以使用普通管理员用户和普通用户的全部功能。并可以进行系统管理，包括高校的各个部门，人员的职位与信息的管理，以及用户添加、修改、删除以及权限的管理。同时还可以查看用户的详细操作日志。

对于不同的用户，系统在任何操作之前都会对其权限进行盘点，权限不足则会得到提示。游客的权限最低，只可以扫码查看资产的基本信息。

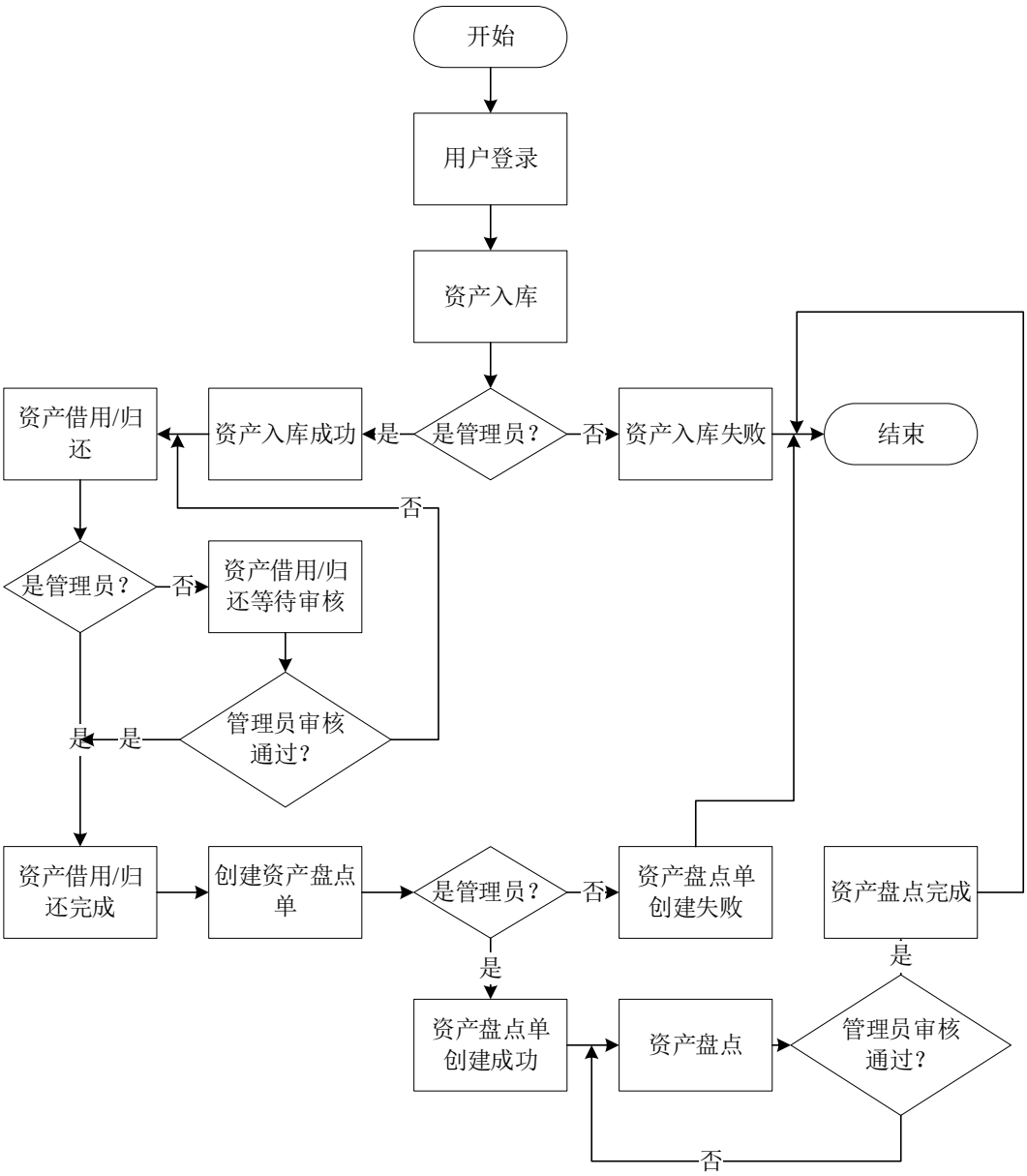

图 3-1 业务流程图

3.1.2 系统需求定义

根据系统的业务流程分析，本系统主要由四类角色构成：游客，高校普通用户，高校普通管理员用户，高校系统管理员用户。系统的用例图如图 3-2 所示。

(1) 游客

一般的游客指的是未登录用户，在本系统中游客除了未登录用户还可以指未绑定在职人员的用户，游客只可以使用 APP 扫码查看资产基本信息，不可以登录

后台。

(2) 高校普通用户

高校普通用户指的是普通学生和教师，就是在高校中并没有资产管理的人员，该类人员为普通用户，普通用户处理可以查看资产基本信息外，还可以进行资产的借用与归还，在接到资产盘点的任务后，进行资产盘点，但所进行的操作都需要管理员进行审核才可以通过。

(3) 高校普通管理员用户

高校普通管理员用户指的是在高校中有资产进行管理的人员，或被赋予资产管理权限的人员，普通管理员用户在普通用户的基础上，还可以资产入库，即资产的增删改查，可以进行盘点单的创建。

(4) 高校系统管理员用户

高校系统管理员用户指的是整个高校固定资产管理系统的管理者，在普通管理员用户的基础上，还可以记性部门管理、职位管理、人员管理、用户管理等操作。

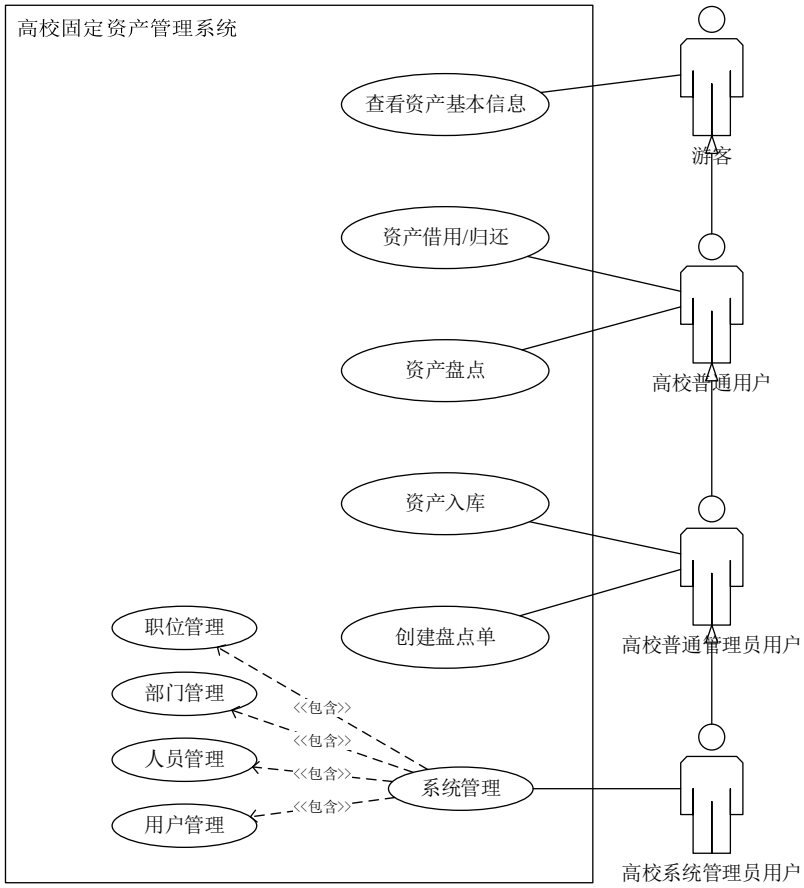

图3-2 系统用例图

3.2 系统总体设计

系统总体设计是对整个系统的设计与实现进行合理的安排，下面将从部署结构设计和系统结构设计进行说明。

3.2.1 部署结构设计

系统基于 B/S 结构设计，分为 PC 端和移动端，PC 端采用 MVC 设计方式，移动端则是 Vue 单页应用。PC 端和移动端分别有不同服务端和客户端，部署关系如图 3-3 所示。其中 Apache 服务器使用 PHP 语言编写，用于处理 PC 客户端的请求，Tornado 服务器采用 Python 语言开发，作为移动客户端的服务端，小程序客户端和手机 APP 客户端与 Tornado 服务器通过 WebSocket 连接。

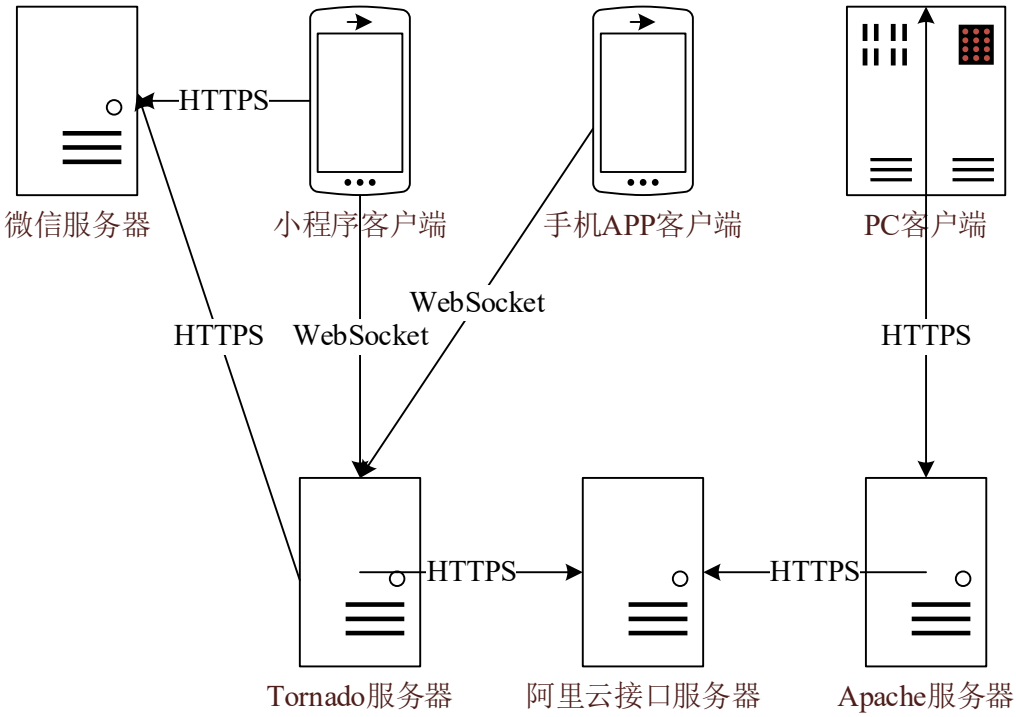

图 3-3 部署关系图

3.2.2 系统结构设计

高校固定资产管理系统分为 PC 后台管理、移动客户端两部分。

(1) PC 后台管理

PC 后台管理采用 Apache Web 服务器进行部署，使用 ThinkPHP5 框架 MVC 设计模式，如图 3-4 所示。主要分为 Public 和 Application 两部分，Public 部分包含

Application 的启动接口以及静态文件的存放，Application 部分包含 Controller、View、Model，Controller 层用于处理逻辑业务，View 层用于数据的显示，Model 层用于数据库数据的处理。

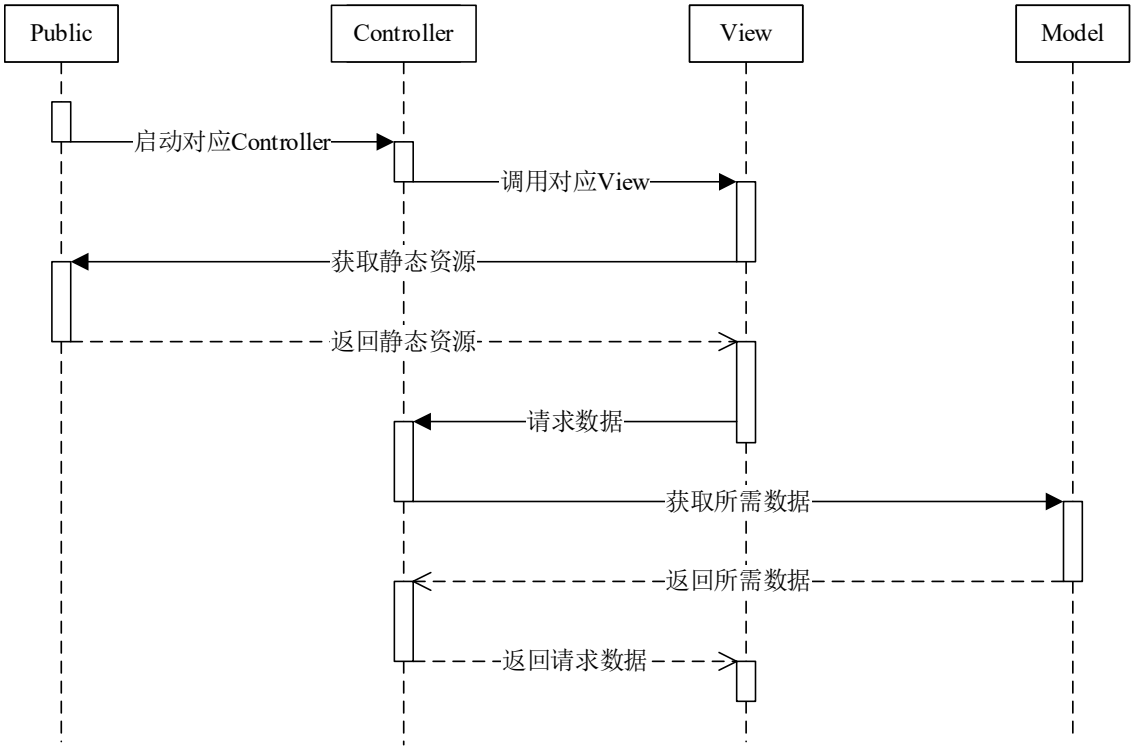

图 3-4 PC 后台管理顺序图

(2) 移动客户端

移动客户端使用 uni-app 编写，每个页面对应一个 Vue 文件，与 Tornado 服务端建立 WebSocket 连接，如图 3-5 所示。Tornado 服务端与移动客户端进行数据交互，并利用 Apache 服务器进行数据处理，移动客户端通过 5+Plus 调用 Android 以及 iOS 接口。

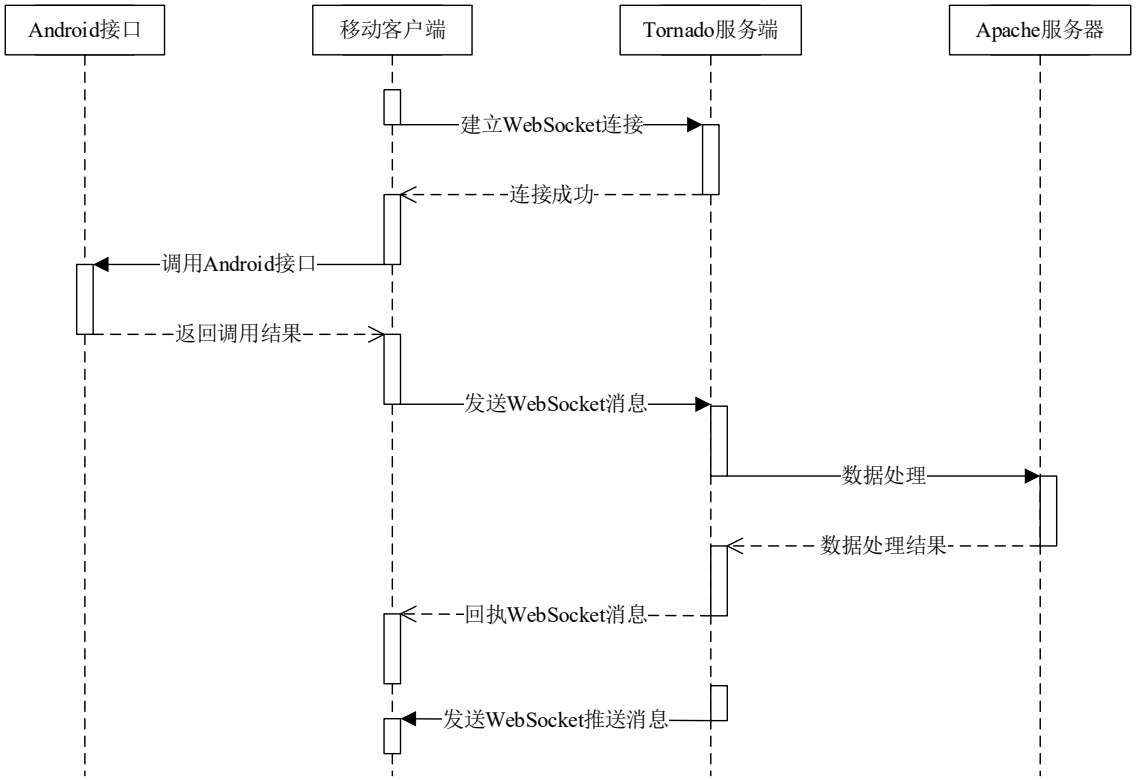

图 3-5 移动客户端顺序图

3.3 数据库设计

本系统中使用 MySQL 数据库为高校固定资产管理系统提供数据存储服务，主要包括资产管理、借用归还、资产盘点、部门管理、职位管理、人员管理、用户管理、日志管理、帖子管理等，下面对一些主要的表进行介绍。

(1) 资产信息表（fams_asset）

资产信息表用于存放资产的信息，具体结构如表 3-1 所示。

表 3-1 资产信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	as_id	int	10	主键	主键
2	as_no	varchar	64	资产编号	唯一
3	as_name	varchar	64	资产名称	非空
4	as_price	double	10,2	资产价格	非空
5	cate_id	int	11	类别id	非空

续表 3-1

序号	数据库字段	字段类型	字段长度	存储内容	备注
6	sta_no	varchar	255	状态编号	非空
7	as_import_time	datetime	0	资产入库时间	非空
8	as_image	varchar	255	资产照片	非空
9	as_local_id	int	11	资产地点	非空
10	as_qrcode	varchar	255	资产二维码	非空
11	as_exist	int	11	记录是否存在	非空

(2) 借用归还表（fams_borrowlend）

借用归还表用于存放资产借用以及归还的相关信息，具体结构如表 3-2 所示。

表 3-2 借用归还表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	bl_id	int	10	主键	主键
2	as_no	varchar	64	资产编号	非空
3	u_id	int	11	用户id	非空
4	b_time	datetime	0	领用时间	非空
5	l_time	datetime	0	归还时间	非空
6	bl_ok	int	11	该记录已完成	非空
7	bl_exist	int	11	记录是否存在	非空

(3) 资产盘点表（fams_check）

资产盘点表用于存放资产盘点单，不存放资产盘点的内容，用于记录盘点单创建和截止的日期以及盘点状态，具体结构如表 3-3 所示。

表 3-3 资产盘点表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	c_id	int	10	主键	主键
2	c_title	varchar	255	盘点单名称	非空
3	u_id	int	11	指定盘点人	非空
4	c_time	datetime	0	盘点单创建日期	非空

续表 3-3

序号	数据库字段	字段类型	字段长度	存储内容	备注
5	c_e_time	datetime	0	盘点限期	非空
6	c_sta_no	varchar	16	盘点单状态	非空
7	c_exist	int	11	记录是否存在	非空

(4) 资产盘点单表（fams_check_content）

资产盘点单表用于存放资产盘点的数据，记录每一个资产的盘点状态，具体结构如表 3-4 所示。

表 3-4 资产盘点单表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	c_c_id	int	10	主键	主键
2	c_id	int	11	所属盘点单id	非空
3	as_no	varchar	64	资产编号	非空
4	c_c_image	varchar	255	盘点图片	非空
5	c_c_sta_no	varchar	16	盘点状态编号	非空
6	c_c_exist	int	11	记录是否存在	非空

(5) 部门信息表（fams_department）

部门信息表用于存放高校各个部门的结构，具体结构如表 3-5 所示。

表 3-5 部门信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	dep_id	int	10	主键	主键
2	dep_no	varchar	32	部门编号	唯一
3	dep_name	varchar	40	部门名称	非空
4	up_dep_id	int	11	上级部门id	非空
5	dep_remark	varchar	255	备注	可空
6	dep_exist	int	11	记录是否存在	非空

(6) 地点信息表（fams_local）

地点信息表用于存放高校各个部门地点的GPS数据，具体结构如图3-6所示。

表 3-6 地点信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	local_id	int	10	主键	主键
2	local_name	varchar	255	地点名称	非空
3	local_data	varchar	255	GPS相关数据	非空
4	local_exist	int	11	记录是否存在	非空

(7) 日志信息表（fams_log）

日志信息表用于存放用户的操作，便于管理员查看，具体结构如表3-7所示。

表 3-7 日志信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	log_id	int	10	主键	主键
2	log_category_no	varchar	32	日志类别编号	非空
3	log_action_no	varchar	32	日志操作编号	非空
4	u_id	int	11	用户id	非空
5	log_datetime	datetime	0	操作时间	非空
6	common_id	int	11	公用id	非空
7	table_name	varchar	64	表名	非空
8	table_main_key	varchar	32	关联的表的主键	非空
9	log_exist	int	11	记录是否存在	非空

(8) 人员信息表（fams_person）

人员信息表用于存放高校的在职人员信息，如学生和教师，具体结构如表3-8所示。

表 3-8 人员信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	p_id	int	10	主键	主键
2	p_no	varchar	32	工号	唯一
3	dep_id	int	11	部门主键	非空
4	pos_id	int	11	职位主键	非空
5	p_name	varchar	40	姓名	非空
6	p_sex	char	2	性别	非空
7	p_ic	varchar	20	身份证号	非空
8	p_email	varchar	40	邮箱	非空
9	p_exist	int	11	记录是否存在	非空

(9) 职位信息表（fams_position）

职位信息表示用于存放高校各个职位的名称以及标号，具体结构如表 3-9 所示。

表 3-9 职位信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	pos_id	int	10	主键	主键
2	pos_no	varchar	32	职位编号	唯一
3	pos_name	varchar	40	职位名称	非空
4	pos_remark	varchar	255	备注	可空
5	pos_exist	int	11	记录是否存在	非空

(10) 资产维修表（fams_repair）

支持维修表用于存放资产维修的信息，具体结构如表 3-10 所示。

表 3-10 资产维修表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	rep_id	int	10	主键	主键
2	as_no	varchar	64	资产编号	非空
3	rep_price	double	10,2	维修费用	非空
4	rep_time	datetime	0	维修时间	非空

续表 3-10

序号	数据库字段	字段类型	字段长度	存储内容	备注
5	rep_exist	int	11	记录是否存在	非空

(11) 用户信息表（fams_user）

用户信息表用于存放用户的信息，具体结构如表 3-11 所示。

表 3-11 用户信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	u_id	int	10	主键	主键
2	p_id	int	11	人员表主键	非空
3	u_phone	varchar	16	用户电话	非空
4	u_openid	varchar	64	微信openid	非空
5	power_no	varchar	16	用户权限编号	非空
6	u_head	varchar	255	用户头像	非空
7	u_money	int	11	用户积分	非空
8	u_login_code	varchar	255	用户登录码	非空
9	u_exist	int	11	记录是否存在	非空

(12) 帖子信息表（fams_bbs）

帖子信息表用于存放移动端首页所发布帖子的信息，具体结构如表 3-12 所示。

表 3-12 帖子信息表结构

序号	数据库字段	字段类型	字段长度	存储内容	备注
1	bbs_id	int	10	主键	主键
2	u_id	int	11	用户id	非空
3	up_bbs_id	int	11	上级id	非空
4	bbs_text	text	0	帖子内容	非空
5	bbs_time	datetime	0	发布时间	非空
6	bbs_exist	int	11	记录是否存在	非空

第 4 章　核心算法及模块的设计与实现

4.1 基于 AES 和 RSA 的二维码加密算法

4.1.1 算法的设计

基于 AES 和 RSA 的混合二维码加密算法流程图如图 4-1 所示，核心内容是对于每一个二维码的内容使用不同 AES 秘钥进行 AES 加密，而这个 AES 秘钥也会进行 RSA 加密放置于二维码中，最后得到加密二维码。

为了保证 AES 秘钥的随机性，将会对每一个 AES 秘钥进行随机轮数的生成组合排列，确保无法被暴力破解，然后利用得到的 AES 秘钥对二维码内容进行 AES 加密，得到加密后的二维码内容，再将 AES 秘钥使用 RSA 私钥进行加密得到加密后的 AES 秘钥，再对加密后的二维码内容以及加密后的 AES 秘钥进行 base64 编码，然后再将得到的两个 base64 编码序列化成 json 字符串，最后将 json 字符串再次进行 base64 编码后生成加密后的二维码。

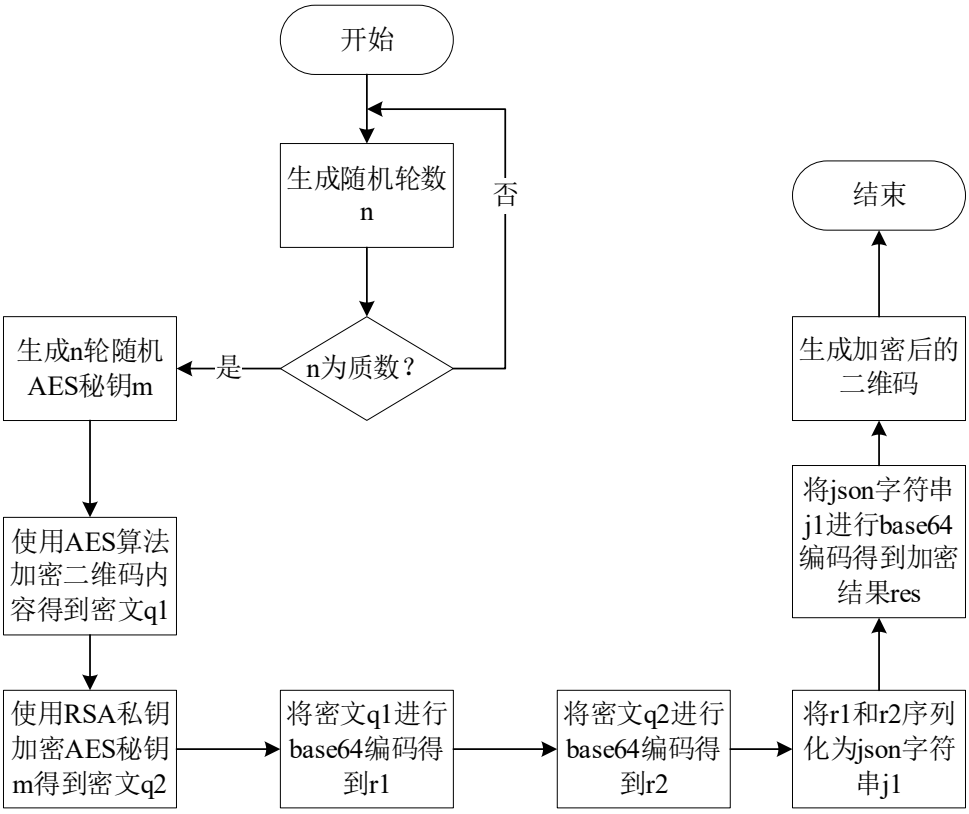

图 4-1　二维码加密算法流程图

基于 AES 和 RSA 的混合二维码解密算法流程图如图 4-2 所示，过程基本就是将加密的过程进行逆向操作，重点在于对于随机轮数的判断，如果随机轮数不是质数或者随机轮数不存在则说明二维码内容被篡改。

算法现将密文进行 base64 反编码，再判断随机轮数是否为质数，是质数则继续进行 base64 反编码，得到 AES 秘钥的密文和二维码内容的密文，再使用 RSA 的公钥对 AES 秘钥的密文进行 RSA 解密得到 AES 秘钥，最后使用 AES 秘钥对二维码内容的密文进行 AES 解密的到二维码明文。

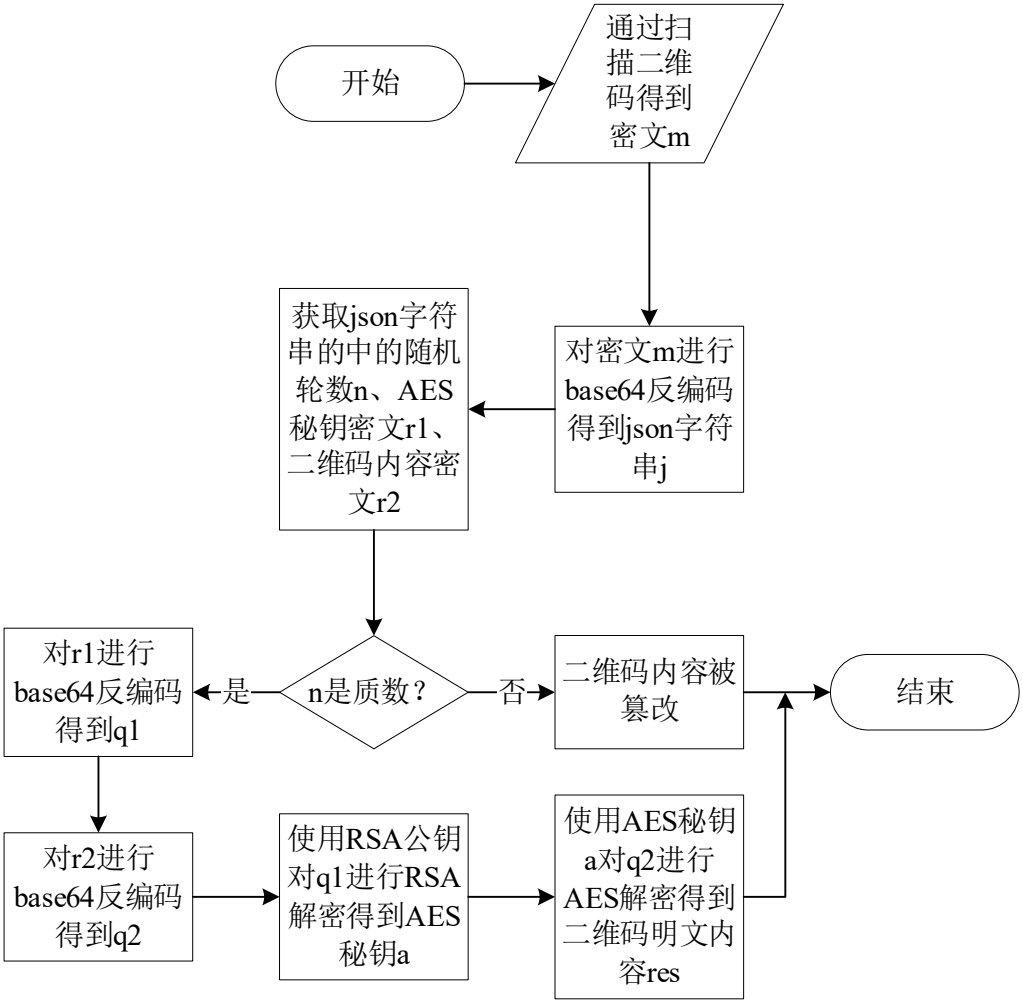

图 4-2 二维码解密算法流程图

4.1.2 算法的实现

该算法有两部分，分别为加密算法和解密算法，由于二维码的生成是在 PC 端进行，所以使用 PHP 语言实现加密算法，而扫描二维码是在移动端进行，所以使用 Python 语言实现解密算法。

(1) 加密算法

加密算法首先需要生成一个随机整数 n，该步骤只需要设定一个随机数种子后生成随机整数即可，而后生成随机 AES 秘钥，代码如下：

```
private function createAESm() {
    $no = "";
    $pattern = '1234567890abcdefghijklmnopqrst
                            uvwxyzABCDEFGHIJKLOMNOPQRSTUVWXYZ';
    $returnStr = '';
    for($i = 0; $i < rand(10,100); $i ++) {
        $returnStr .= $pattern[mt_rand(0, 61)];
    }
    $no = $returnStr . time();
    $returnStr = '';
    for($i = 0; $i < rand(10,100); $i ++) {
        $returnStr .= $pattern[mt_rand(0, 61)];
    }
    $no .= $returnStr;
    return strtoupper(md5($no));
}
```

上述代码中$no 是即将生成的秘钥，$pattern 是可能为秘钥内容的字符串，通过 for 循环随机生成长度，随机选取字符，再与时间戳记性拼接，最后进行 md5 编码，strtoupper()方法是将字符串中所有字符转换为大写，最后的到 AES 秘钥。

生成秘钥后使用得到 AES 秘钥对二维码内容进行加密，代码如下：

```
private function aesEncode($data, $aesM){
    $method = 'AES-128-ECB';
    $options = 0;
    $iv = '';
    return openssl_encrypt($data, $method, $aesM, $options, $iv);
}
```

上述代码中，$method 代表加密方式，AES 一种有 5 种加密方式：电码本模式（Electronic Codebook Book (ECB)）、密码分组链接模式（Cipher Block Chaining (CBC)）、计算器模式（Counter (CTR)）、密码反馈模式（Cipher FeedBack (CFB)）、输出反馈模式（Output FeedBack (OFB)），本文选用的是 ECB 电码本模式（AES-128-ECB）其中 128 代表秘钥为 16 位，但生成并且记录的秘钥为 32 位，多出的 16 位不参与算法，其迷惑作用，$options 是以下标记的按位或，$iv 是非 NULL 的初始化向量，$data 为待加密数据，$aesM 为秘钥，openssl_encrypt()为 AES 加密的方法。

对明文进行加密后使用提前生成好的 RSA 私钥对 AES 秘钥进行加密，代码如下：

```
private function rsaEncode($data, $rsaPrivateKey) {
    $search = [
        "-----BEGIN RSA PRIVATE KEY-----",
        "-----END RSA PRIVATE KEY-----",
        "\n",
        "\r",
        "\r\n"
    ];
    $rsaPrivateKey = str_replace($search, "", $rsaPrivateKey);
    $privateKey = $search[0] . PHP_EOL . wordwrap(
        $rsaPrivateKey, 64, "\n", true) . PHP_EOL . $search[1];
    openssl_private_encrypt($data, $rst, $privateKey);
    return $rst;
}
```

上述代码中，$data 为待加密数据，$rsaPrivateKey 为 RSA 私钥字符串，$search 为数组，用于存放私钥的数据，使用$search 和$rsaPrivateKey 组合成完整的私钥 $privateKey，openssl_private_encrypt()为 RSA 加密的方法.

对 AES 秘钥进行加密后对两个密文进行 base64 编码再组合成 json 字符串再次进行 base64 编码，得到最终结果，代码如下：

```
private function createResult($q1, $q2, $n) {
    $jsonData = [
        "n" => $n,
        "r1" => base64_encode($q1),
        "r2" => base64_encode($q2)
    ];
    $jsonString = json_encode($jsonData);
    $result = base64_encode($jsonString);
    return $result;
}
```

上述代码中，$q1 为 AES 秘钥的密文，$q2 为二维码内容的密文，$n 为随机轮数，将$q1 与$q2 使用 base64_encode()方法进行编码后与$n 组合成关联数组，再使用 json_encode()转换成为 json 字符串，再次进行 base64 编码后得出最终结果。

得到结果后将得到的结果生成二维码，代码如下：

```
public static function createQrcode($value, $name) {
    vendor('phpqrcode');
    $errorCorrectionLevel = 'L';
```

```
        $matrixPointSize = 5;
        $filename = 'qrcode/'.$name.'.png';
        \QRcode::png($value, $filename,
                $errorCorrectionLevel, $matrixPointSize, 2);
    }
```

上述代码中，$value 是二维码内容，$name 是文件名称，首先使用 vendor()方法加载 phpqrcode 模块，$errorCorrectionLevel 控制二维码容错率，包括 L(QR_ECLEVEL_L，7%)、M(QR_ECLEVEL_M，15%)、Q(QR_ECLEVEL_Q，25%)、H(QR_ECLEVEL_H，30%)，本文采用最低的 L，$matrixPointSize 控制生成图片的大小，$filename 是文件路径及文件名 QRcode::png 是调用类 QRcode 中的静态方法 png 用于生成并保存二维码图片。

(2) 解密算法

解密算法需要先从密文中提取出随机轮数、密文 AES 秘钥、密文二维码内容，代码如下：

```
def loadData(data):
    jsonString = base64.b64decode(data)
    jsonData = json.loads(jsonString)
    n = jsonData['n']
    q1 = base64.b64decode(jsonData['r1'])
    q2 = base64.b64decode(jsonData['r2'])
    return n, q1, q2
```

上述代码中，data 为密文数据，使用 base64.b64decode()函数将密文进行 base64 反编码得到其中的 json 字符串，再使用 json.loads()函数对 json 字符串进行反编码得到 json 对象，在这里为一个字典。

得到密文中的数据后，判断随机轮数是否为质数，确定二维码是否被篡改，再使用提前生成好的 RSA 公钥对 AES 秘钥的密文进行 RSA 解密，代码如下：

```
def rsaDecode(data, rsaPublicKey):
    content = rsa.decrypt(data, rsaPublicKey)
    rst = content.decode("utf-8")
    return rst
```

上述代码中，使用了 Python 中的 rsa 模块，rsa.decrypt()函数是用于 RSA 解密，其中 data 为密文，resPublicKey 是 RSA 的公钥，这里的公钥是 rsa 模块中的公钥对象，不是一个字符串。

进行 RSA 解密后，得到 AES 秘钥，使用刚得到了 AES 秘钥对二维码密文进行 AES 解密，代码如下：

```
def aesDecode(data, aesM):
    aes = AES.new(aesM, AES.MODE_ECB)
    base64Dec = base64.decodebytes(data.encode(encoding='utf-8'))
    rst = str(aes.decrypt(base64Dec), encoding='utf-8')
    return rst
```

上述代码中，使用了 Python 中的 Crypto 模块，data 为密文，aesM 为 AES 秘钥，AES.new()创建一个 AES 对象，采用 ECB 方式，ECB 在这里的表示为 AES.MODE_ECB，使用 base64.decodebytes()函数将密文转换成为 base64 字节对象，不可以使用字符串，最后使用 aes.decrypt()函数进行 AES 解密得到最终的明文。

4.2 三通道颜色直方图的资产图片核对算法

4.2.1 算法的设计

三通道颜色直方图的资产图片核对算法流程图如图 4-3 所示，核心内容是通过巴氏系数公式计算 RBG 三通道的颜色直方图数据的相似度的平均值，通过大量数据得到资产核对所需要的阈值。算法采用 Python 语言编写，只用到了计算机视觉库 OpenCV。

首先读取图片并重置两图片的尺寸，确保两图片尺寸相同，这样得到的直方图数据的长度就相同，然后对两图片进行 RGB 通道的拆分，再分别对三通道的数据进行计算，最后取平均值作为结果。

单通道相似度计算需要先获取两图片的颜色直方图数据，对于重合的数据设置为 1，不重合的数据使用巴氏系数公式进行计算，最后得到单通道的相似度。

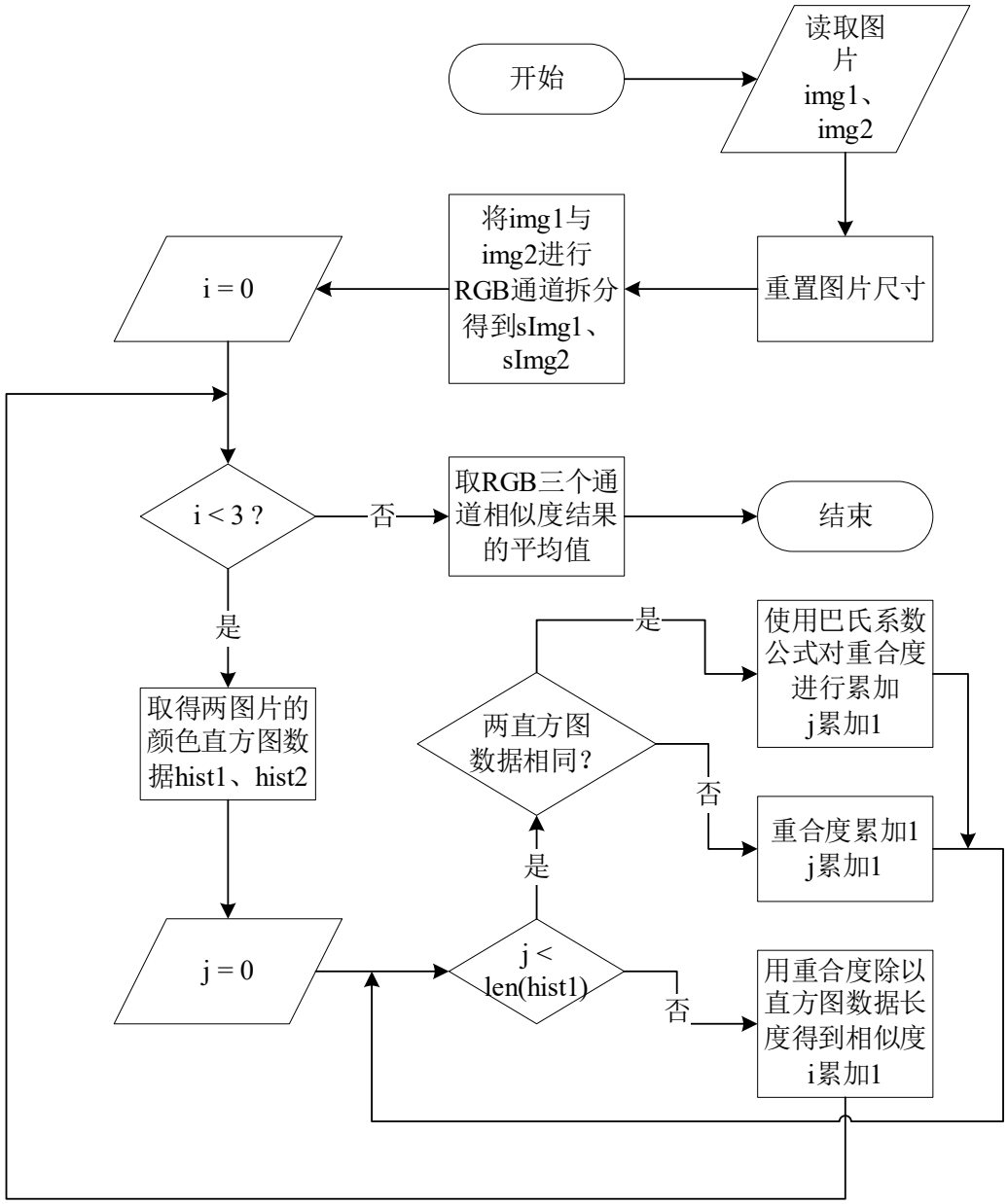

图 4-3 资产图片核对算法流程图

4.2.2 算法的实现

该算法是在使用移动端进行资产盘点时使用，所以使用 Python 语言借助 OpenCV 实现该算法。

首选需要读取两张图片，代码如下：

```
def openImage(path1, path2):
    size=(256, 256)
```

```
    image1 = cv2.imread(path1)
    image2 = cv2.imread(path2)
    image1 = cv2.resize(image1, size)
    image2 = cv2.resize(image2, size)
    return image1, image2
```

上述代码中，cv2.imread()函数用于读取本地图片，cv2.resize()函数用修改图片资产，这里为了统一两图片的资产，因为上传的图片都需要裁剪为正方形，所以本文统一尺寸为(256, 256)。

读取完图片后，对 RGB 通道进行拆分，代码如下：

```
def splitImage(image1, image2):
    image1 = cv2.split(image1)
    image2 = cv2.split(image2)
    return image1, image2
```

上述代码中，cv2.split()函数的作用是对图片进行 RGB 通道的分割。

对通道进行分割后，计算每个通道的相似值，计算单通道相似值的代码如下：

```
def calculate(image1, image2):
    hist1 = cv2.calcHist([image1], [0], None, [256], [0.0, 255.0])
    hist2 = cv2.calcHist([image2], [0], None, [256], [0.0, 255.0])
    degree = 0
    for i in range(len(hist1)):
        if hist1[i] != hist2[i]:
            degree = degree + (1 - abs(hist1[i] -
                        hist2[i]) / max(hist1[i], hist2[i]))
        else:
            degree = degree + 1
    degree = degree / len(hist1)
    return degree
```

上述代码中，cv2.calcHist()函数是获取颜色直方图数据，然后利用 for 循环对每个数据进行计算，得到相似值。

最后取三通道相似值的平均数，代码如下：

```
def calcResult(image1, image2):
    result = 0
    for im1, im2 in zip(image1, image2):
        result += calculate(im1, im2)
    result = result / 3
    return result
```

4.3 MacOS 风格弹出层组件的设计与实现

4.3.1 MacOS 风格弹出层组件的设计

ljspopup 是仿 Mac OS 窗口风格的弹出层组件，采用 JavaScript 语言基于 jQuery 编写，弹出后的样式如图 4-4 所示。

图 4-4 弹出层界面

可以在同一个页面弹出多个弹出层，后弹出的弹出层会在先弹出的上层，如图 4-5 所示。

图 4-5 多个弹出层

弹出层可以进行移动，将鼠标指向标题，按住鼠标拖动即可移动弹出层位置，弹出层可以最大化、最小化，Mac OS 窗口风格，弹出层左上角有红、橙、绿三种颜色的按钮，红色按钮为关闭弹出层，橙色按钮为最小化弹出层，绿色按钮为最大化弹出层，如图 4-6 所示。

图 4-6 最小化弹出层

默认根据内容调整弹出层调整大小，弹出层内容为文字、HTML 内容或元素 id。

而弹出层的弹出、关闭、最小化、最大化等操作，是通过 jQuery 自定义动画实现，弹出首选设置弹出层的宽和高都为 0，然后再扩大到预设宽高的 110%，最后恢复到预设的宽高，这样就实现了动态的弹出效果，关闭则是将宽高直接缩小为 0，而最小化则是先计算出最小化的位置再将宽高以及位置通过 jQuery 动画缩小，实现动态的最小化，而这些操作都可以禁用，如图 4-7 是禁用最大化、最小化、移动以及不显示遮罩层的效果图。

图 4-7 禁用效果图

4.3.2 MacOS 风格弹出层组件的实现

弹出层只有 ljspopup.js 和 ljspopup.css 两个文件，JS 文件用于编写弹出层的逻辑算法，CSS 文件用于编写弹出层样式。

因为可以弹出多个弹出层，所以定义数组 popup_data 用于存储每一个弹出层数据，这样就可以记录已经创建的弹出层的个数，避免弹出层的重复创建，浪费系统资源造成卡顿，而且还可以记录弹出层的位置信息和大小，方便最大化、最

小化、还原时的数据初始化，调用弹出层是的参数为一个字典，所以弹出的函数定义应为：

```
function ljspopup(data = {}){}
```

data 即为传入的参数，默认为空字典，即为缺省数据，一般调用时会传入 el、title 等参数，调用此函数并传入参数，首先创建该弹出层的 sign，也就是唯一编号，并设置在 id 中，用于辨认弹出层，并判断该弹出层是否已经被创建，代码如下：

```
var panel_sign = "ljs-sign-" + $.md5(JSON.stringify(data));
var index = sign_find(panel_sign);
```

如果弹出层已经被创建则直接显示弹出层即可，如该弹出层没有被创建，则应先创建弹出层，要想创建弹出层，首先需要对参数进行初始化，确保每个参数都有一个缺省值，可以传入的参数：el（弹出层显示的元素选择器或 HTML 代码）、title（弹出层标题）、width（弹出层宽度）、height（弹出层高度）、top（弹出层距离文档顶部距离）、left（弹出层距离文档左侧距离）、min（是否允许最小化）、max（是否允许最大化）、mask（是否显示遮罩层）、move（是否允许移动）。

参数初始化结束后创建弹出层所需的元素，即创建 DOM，使用原生 JS 中的 document.createElement()方法创建，并保存在相应变量中，部分代码如下：

```
var ljs_panel = document.createElement("div");
var ljs_panel_header = document.createElement("div");
```

创建好元素后，再对已经创建好的弹出层元素赋予 CSS 样式，利用 jQuery 将 DOM 对象转换为 jQuery 对象，并使用 jQuery 中的 addClass()方法对其添加预设好的 CSS 样式，部分代码如下：

```
$(ljs_panel).addClass("ljs-panel");
$(ljs_panel).addClass(panel_sign);
```

赋予 CSS 样式后，对弹出层元素赋予内容并添加到<body>中，利用 jQuery 中的 html()方法设置 DOM 对象的 innerHTML 属性，并使用 jQuery 中的 append()方法代替 DOM 对象的 appendChild 方法向文档中添加 DOM 元素，部分代码如下：

```
$(ljs_panel_header_close).html("×");
$(ljs_panel_header_min).html("-");
$(ljs_panel_body).append($(new_data.el));
$("body").append(ljs_panel);
```

然后对弹出层的位置进行调整，并添加到数组 popup_data 中，再判断是否显

示遮罩层，首先查看<body>中是否存在遮罩层元素，如果不存在则创建遮罩层元素并添加到<body>中，然后根据弹出层的属性判断是否显示遮罩层，代码如下：

```
if (!el_exist(".mask")) {
    var mask = document.createElement("div");
    $(mask).addClass("mask");
    $("body").append(mask);
}
if (data.mask) {
    $(".mask").show();
}
```

上诉代码中，el_exist()方法的作用是判断元素是否在文档中。

然后读取弹出层配置，弹出层的配置即使传入的参数 data，读取配置用于计算弹出动画的元素位置等信息，最后以 jQuery 动画的方式将弹出层显示出来，代码如下：

```
$panel.css({
    width: 0,
    height: 0,
    top: top,
    left: left,
    display: "flex"
}).animate({
    width: "+=" + (width + width_out) + "px",
    height: "+=" + (height + height_out) + "px",
    top: "-=" + (height + height_out) / 2 + "px",
    left: "-=" + (width + width_out) / 2 + "px"
}, 150).animate({
    width: "-=" + width_out + "px",
    height: "-=" + height_out + "px",
    top: "+=" + height_out / 2 + "px",
    left: "+=" + width_out / 2 + "px"
}, 100);
```

上诉代码中，css()方法是对元素初始 CSS 样式进行设置，animate()方法是执行动画，在规定时间内将 CSS 样式变为指定样式。

至此，弹出层显示完毕，然后需要进行事件监听，利用 jQuery 中的 on()方法进行事件监听，例如最小化弹出层，需要在网页最下面创建一个区域用于存放最小化的弹出层，采用从右到左，从下到上的方式排列，方法与遮罩层相似，然后，根据已有最小化弹出层数量和屏幕的宽度计算此弹出层需要取得绝对位置，为最小化动画做准备，关键代码如下：

```
var now_number = $(".ljs-panel-min-area").children().length;
var max_number = Math.floor(window_width / min_width);
var c = now_number / max_number;
var now_line = c < Math.floor(c) ? Math.floor(c) :
                                        Math.floor(c) + 1;
var now_col = (now_number % max_number) + 1;
```

上述代码中，Math.floor()方法为向下取整。

最后，进行 jQuery 动画，再将此弹出层加入底部区域。完成后对外提供两个可主动调用的接口：var p = ljspopup({})（显示弹出层，参数为上述所介绍参数）、closeLjspopup(p)（关闭弹出层，参数为弹出层对象）。

4.4 Python 数据库访问类的设计与实现

4.4.1 Python 数据库访问类的设计

在编写 Python 代码的过程中对数据的操作必不可少，对于一个项目，如果涉及到大量 SQL 语句，会导致程序的可读性变差，出现 bug 也不容易排查，所以封装一个好的数据库访问类尤为重要。

Ljsmysql 是使用 Python 语言基于 pymysql 编写的 MySQL 数据库访问类，无需编写 SQL 语句即可实现数据库增删改查操作。使用 connect 函数进行数据库连接，使用 table 函数选择要操作的表，使用 where 函数设置条件，使用 select、find 函数进行数据查找，使用 insert 函数插入数据，使用 update 函数修改数据，使用 delete 函数删除数据，使用 order 函数进行排序。提供如下对外接口：

(1) 添加记录

```
Ljsmysql.table("表名").insert({
    "字段 1": "值 1",
```

```
    "字段 2": "值 2"
})
```

(2) 删除记录

```
Ljsmysql.table("表名").where("字段", "值").delete()
```

(3) 修改记录

```
Ljsmysql.table("表名").where("字段", "值").update({
    "字段 1": "值 1",
    "字段 2": "值 2"
})
```

(4) 判断记录是否存在

```
Ljsmysql.table("表名").where("字段", "值").find()
```

(5) 查找记录

```
Ljsmysql.table("表名").where("字段", "值").select()
```

(6) 查找记录并排序

```
Ljsmysql.table("表名").where("字段", "值").
                                    order("字段 desc").select()
```

4.4.2 Python 数据库访问类的实现

访问数据库首先需要进行数据库的连接，封装为 connect()函数，代码如下：

```
@staticmethod
def connect(localhost="127.0.0.1", username="root", password="",
                                  dbname="mysql", encoding="utf8"):
    Ljsmysql.db = pymysql.connect(
        localhost, username, password, dbname, charset=encoding)
    Ljsmysql.cursor = Ljsmysql.db.cursor(
                                        pymysql.cursors.DictCursor)
```

上述代码中，修饰@staticmethod 代表此函数为类中的静态函数，pymysql.connect()函数的作用是连接数据库，对应的参数分别是：数据库地址、用户名、密码、数据库名、字符编码，cursor()函数的作用是创建游标，pymysql.cursors.DictCursor 是一个常量，代表使用该游标查询数据返回的结果为字典。

上述对外接口中，table()函数来选择要操作的表，返回一个 Table 对象，where()

函数是条件，与 SQL 语句中 WHERE 对应，where()可传入动态参数，最少一个，最多三个，参数个数不同代表含义也就不同，一个参数可以传入字典或列表，代表多条件查询，两个参数最常见，字段对应值即可，三个参数比两个参数多了字段与值之间的管理，例如：like、not like 等。

生成 SQL 语句，使用 Python 中的 format()函数对字符串进行格式化，首先对 WHERE 语句进行拼接，生成 WHERE 条件语句保存于当前对象中，然后将 ORDER 排序语句保存于当前对象中，最后根据 select()、update()、delete()、insert()不同函数采取对应的 SQL 语句拼接策略，SQL 语句生成后使用 pymysql.escape_string()函数记性自动转义字符，防止 SQL 注入。

第 5 章　高校固定资产管理系统的实现

高校固定资产管理系统分为 PC 端和移动端，而移动端又分为客户端和服务端，下面分别介绍 PC 端和移动端的实现。

5.1 高校固定资产管理系统 PC 端的实现

5.1.1 项目部署及运行环境

高校固定资产管理系统 PC 端采用 PHP 语言编写，ThinkPHP5 框架。MVC 的设计模式，在 Apache Web 服务器上运行，部署 SSL 证书，通过 443 端口的 HTTPS 协议进行访问。

5.1.2 主要功能实现

高校固定资产管理系统主要分为 4 个功能模块，分别为：资产管理、资产盘点、审批管理和系统管理。

(1) 资产管理

资产管理主要分为资产入库以及借用归还两部分，资产入库主要是进行资产信息的录入，首先填写资产的基本信息，如图 5-1 所示。

图 5-1 填写资产基本信息

随后进行资产图片的上传，由于不同手机或相机拍摄的图片大小不同，所以需要进行裁剪，如图 5-2 所示。

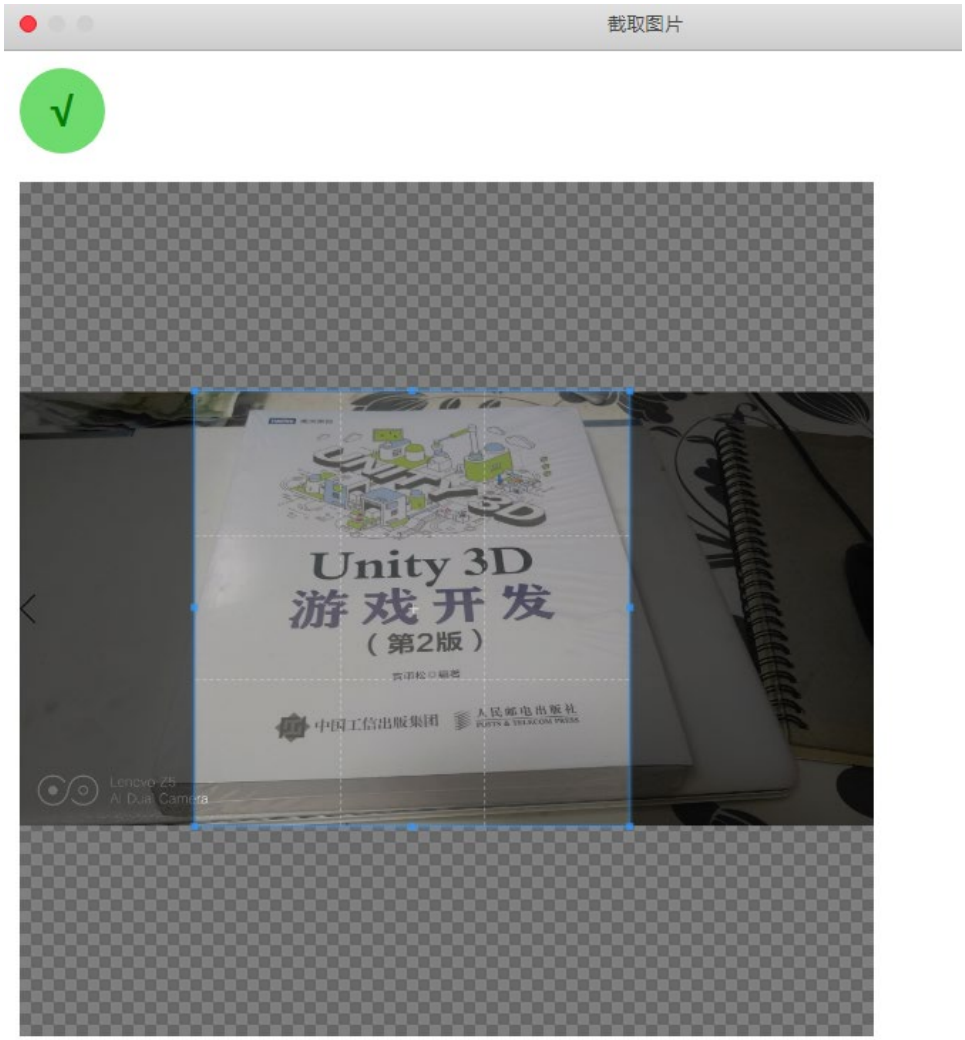

图 5-2 截取图片

确定裁剪区域后，将裁剪区域的数据和图片的地址发送给后台，后台进行裁剪并将新图片地址返回给前端，实现代码如下：

```
public function crop() {
    $img = Param::get("img");
    $x = Param::get("x");
    $y = Param::get("y");
    $width = Param::get("width");
    $height = Param::get("height");
    $path = ROOT_PATH . 'public' . DS . "image" . DS . $img;
```

```
        $src = imagecreatefromstring(file_get_contents($path));
        $new_image = imagecreatetruecolor($width, $height);
        $ext = pathinfo($path, PATHINFO_EXTENSION);
        $rand_name = md5(mt_rand() . time()) . "." . $ext;
        imagecopyresampled($new_image, $src, 0, 0, $x, $y, $width,
                           $height, $width, $height);
        imagejpeg($new_image, ROOT_PATH . 'public' . DS . "image" . DS .
                           "crop" . DS . $rand_name);
        imagedestroy($src);
        imagedestroy($new_image);
        return "/crop/" . $rand_name;
    }
```

上述代码中，Param 为参数获取的静态类，使用 get()方法获取 Request 中的参数，$path 为待裁剪图片路径，imagecreatefromstring()方法是将字符串转换成 PHP 图片对象，file_get_contents()方法是把整个文件读取到字符串中 imagecreatetruecolor()方法是创建一个空白的 PHP 图片对象，pathinfo()方法用于获取文件信息，PATHINFO_EXTENSION 参数为常量，作用为只获取文件扩展名，mt_rand()和 time()方法是生成随机数和获取当前时间戳，imagecopyresampled()是将一张图片的某一区域复制到另一张图片上，imagejpeg()方法是将 PHP 图片对象保存成图片文件，imagedestroy()方法是用来释放内存的，将 PHP 图片对象销毁。

获取到裁剪后的图片路径后，将其和之前填写好的资产基本信息发送给后台存入数据库即完成了资产入库。

资产的借用与归还是两个操作，资产的借用需要填写借用的基本信息，普通用户只可以自己借用资产，而管理员则可以指定借给谁资产，如图 5-3 所示。

图 5-3 资产借用

然后将信息传输给后台，后台只需进行数据库的增删改查操作即可。

(2) 资产盘点

资产盘点的具体功能需要在移动端完成，PC 端只负责创建盘点单即可，创建盘点单，需要填写盘点单的名称、截止时间等信息，如图 5-4 所示。

图 5-4 创建盘点单

可以指定盘点人员，勾选添加盘点人员领用的资产到盘点单，可以将该人员借用的资产添加到盘点单中，点击创建后将数据发送到后台，后台进行盘点单的创建，实现代码如下：

```
public function insert() {
    if (Utils::userAlreadyLogin() && $this->powerTrue()) {
        $co = $this->checkOnlyAndNull();
        if ($co['code'] == 0) {
            return json_encode($co);
        }
        $data = $this->getFields();
        $data['c_time'] = date("Y-m-d H:i:s");
        $commonId = Check::insert($data);
        if ($data['u_id'] > 0 && Param::get("checked")) {
            $assets = Borrowlend::getAllbyUId($data['u_id']);
            foreach ($assets as $asset) {
                Check_content::insert([
                    "c_id" => $commonId,
                    "as_no" => $asset['as_no']
                ]);
            }
        }
        return json_encode(Utils::returnCode(1));
    } else {
        $this->error(Constant::PLEASELOGIN, Constant::LOGINPATH);
    }
}
```

上述代码中，Utils 为工具类，封装了多种常用的方法，userAlreadyLogin()和powerTrue()方法适用于判断用户是否登录以及用户的权限，checkOnlyAndNull()用于检查 Request 数据中是否存在关键项为空和存在值不合法的问题，getFields()方法是获取 Request 中的数据并转换成适用于数据库字段的关联数组，date()方法是获取当前的日期时间。Check 类似 Java 中的 bean，每一个数据库中的表都有对应

的类，而 Check 就是用于存放盘点单的表对应的类，Check 中的 insert()方法是将数据写入到数据库中，Borrowlend 是存放借用归还信息的表对用的类，getAllbyUId()方法是根据用户 id 获取其借用的所有资产信息，利用 foreach 遍历找到的结果，将其接入到盘点单中。

(3) 审批管理

审批管理只存在于普通用户进行操作，例如借用资产，然后操作进入管理员的审批列表中，管理员即可同意和驳回，后台只需修改资产的状态以及对借用归还表进行增删改查即可。

(4) 系统管理

系统管理则是系统管理员对组织架构、使用用户进行管理、例如部门管理、用户权限修改等，部门管理如图 5-5 所示。

图 5-5 部门管理

图中左侧为部门的组织架构，以无限级 treeview 显示，右侧则是部门的信息，可以对部门信息进行增删改查。

5.2 高校固定资产管理系统移动端的实现

5.2.1 项目部署及运行环境

高校固定资产管理系统的移动端从部署上看分为服务端和客户端，服务端采用 Python 语言编写，在 Tornado Web 服务器上运行，部署 SSL 证书，通过 8888 端口使用 WSS 协议进行 WebSocket 连接，而客户端又分为微信小程序和手机 APP，微信小程序直接发布到微信公众平台，手机 APP 则需要安装在智能手机中，基于手机中自带的 WebView 运行。

5.2.2 主要功能实现

高校固定资产管理系统的移动端和 PC 端功能相近，移动端可以扫描二维码查看资产信息，如图 5-6 所示。

图 5-6 扫描二维码

图 5-7 资产信息

扫描成功后会与服务端进行通讯，验证二维码的有效性，并获取资产的基本信息，如图 5-7 所示。

与 PC 端不同的是，移动端的用户之间可以相互交流，每个用户都可以发布帖

子，询问一些自己不知道的问题，如图 5-8 所示。

图 5-8 发帖

但为了用户间的信息保密，所以用户之间不可以相互查看个人信息，如图 5-9 所示，首页可以查看全部的额帖子。

图 5-9 首页

图 5-10 帖子详情

用户可以挑选感兴趣的话题进行回复，如图 5-10 所示。人多力量大，不是所有问题都需要管理员来进行解决，管理员也可以将一些注意事项发布到帖子里面。

获取帖子信息以及回复消息的客户端代码如下：

```
onLoad: function(options) {
    this.id = options.id
    getApp().sendSocket("selectRecovery", {id: options.id})
}
```

上述代码中，onLoad 为生命周期函数，当页面加载的时候调用，options 为传递到此页面的参数，该页面只传输了一个 id，这是帖子的 id，getApp()方法是获取 App.vue 的实例，sendSocket()方法作用是发送消息到服务端，实现代码如下：

```
sendSocket: function(msg, obj = "") {
    if (this.globalData.socketConnectStatus) {
        let sendContent = JSON.stringify({
            msg: msg,
            obj: obj
        })
        uni.sendSocketMessage({
            data: sendContent
        })
        console.log("sendMessage:", sendContent)
    } else {
        this.reConnectSocket(msg, obj)
    }
}
```

上述代码中，msg 为发送的命令，obj 为传输的数据，socketConnectStatus 是连接的状态，是否与服务端连接，uni.sendSocketMessage()方法是将消息发送到服务端的方法，data 代表要传输的消息，只可以为字符串，如果未连接或连接断开则需要重新连接，方法为上述代码中的 this.reConnectSocket()，实现代码如下：

```
reConnectSocket: function(msg, obj) {
    uni.showLoading({
        title: "正在重新连接..."
    })
```

```
        uni.onSocketOpen(() => {
            uni.hideLoading()
            uni.showToast({
                icon: "success",
                title: "连接成功"
            })
            this.globalData.socketConnectStatus = true
            this.loginSocket()
            setTimeout(() => {
                this.sendSocket(msg, obj)
            }, 1000)
        })
        uni.onSocketError(() => {
            uni.hideLoading()
            uni.showToast({
                icon: "none",
                title: "连接失败"
            })
        })
        uni.closeSocket({
            complete: () => {
                uni.connectSocket({
                    url: this.globalData.socketUrl
                })
            }
        })
    }
```

上述代码分为 5 个部分，第 1 部分 uni.showLoading()用于提示，告知用户正在重新连接，第 2 部分 uni.onSocketOpen()定义连接成功后的方法，loginSocket()自动登录，然后再次调用发送消息的方法，第 3 部分 uni.onSocketError()定连接失败后的方法，第 4 部分 uni.closeSocket()为了防止二次连接，需要先断开连接，然后不

论成功还是失败，直接进入第 5 部分 uni.connectSocket()连接服务端。

接下来就是服务端的操作，获取帖子信息以及回复消息的服务端代码如下：

```
def selectRecovery(tornadoSelf, obj):
    if tornadoSelf.userid and tornadoSelf.userid > 0:
        dbData = list(Ljsmysql.table("fams_bbs").where("bbs_id",
                obj['id']).order("bbs_time desc").select())
        dbData += list(Ljsmysql.table("fams_bbs").
                 where("up_bbs_id", obj['id']).
                 order("bbs_time desc").select())
        sendMsg(tornadoSelf, "recovery", dbData)
```

上述代码中，Ljsmysql 是数据库操作类，table()选取表，where()设置查找的条件，order()用于排序，select()执行查找操作，sendMsg()用于发送消息到客户端。

第6章　结　论

以进一步提高高校固定资产管理水平和能力为目的，设计了高校固定资产管理系统，面向系统管理员、资产管理员以及普通用户提供了系统管理、资产管理、资产的借用与归还以及资产盘点的功能和服务，实现了对高校固定资产管理业务活动的信息化支持。测试结果表明，该系统可以在服务器上稳定运行，达到了预期设计目标。

系统整体上采用了 MVC 的设计模式和原则，建立了高校固定资产管理的总体结构和运行机制，通过前后端分离增加了代码的可读性，增强了系统的可维护性。通过 AES 与 RSA 加密技术实现了基于 AES 和 RSA 的二维码加密算法，验证了关于二维码内容加密问题的一种解决方法；针对不同二维码采用了不同的秘钥进行加密，避免了由秘钥泄露造成的加密算法被破解；通过颜色直方图技术实现了基于三通道颜色直方图的资产图片核对算法，验证了关于资产图片核对问题的一种解决方法；针对图片采用了均值哈希算法、差值哈希算法以及感知哈希算法，避免了由两张图片大部分颜色相同造成的误判断。

目前该系统对资产图片核对的处理仍有较明显的局限性，主要表现为资产进行涂改后无法进行判断，通过运用图像识别的方法和技术将可以进一步提升高校固定资产管理系统在资产图片核对方面的功能和表现，这将是本文工作今后研究和验证的重点内容和主要方向。

参考文献

[1] 张慕然. 高校固定资产管理系统设计与实现[D]. 长江: 长江大学, 2019.

[2] 姜宇飞. 设备资产管理的设计与实现[D]. 青岛: 青岛大学, 2019.

[3] 张华. 基于非对称加密算法的 QR 二维码[J]. 电子技术与软件工程, 2018(05): 29-29.

[4] 高硕. 基于内容的图像检索系统研究与实现[D]. 唐山: 华北理工大学, 2019.

[5] 刘海峰, 刘洋, 梁星亮. 一种结合优化后 AES 与 RSA 算法的二维码加密算法[J]. 陕西科技大学学报, 2019, 37(06): 153-159.

[6] 王宁邦, 徐博, 夏百川, 夏娜, 郃永航, 李琼. 一种颜色直方图计算相似度的资产图片核对算法[J]. 智能计算机与应用, 2018, 8(02): 59-62, 67.

[7] 李先懿, 郭正光. 基于 Websocket 的车联网报警推送系统[J]. 计算机系统应用, 2020, 29(03): 127-131.

[8] 柴青山. 基于 MVVM 模式的 Vue.js 框架在物流软件自动化测试系统中的应用研究[D]. 北京: 北京邮电大学, 2019.

[9] 许溜溜. 基于 HBuilder 快速开发移动端 APP 的设计与实现[J]. 电脑知识与技术, 2020, 16(10): 74-75.

[10] 林国梁. 基于 Web 的移动终端取证管理系统的设计与实现[D]. 厦门: 厦门大学, 2018.

[11] 李全全. Python OpenCv 在智慧党建人脸识别中的应用[J]. 中国有线电视, 2020, (02): 167-171.

[12] 田维铝, 邓梅, 王晓华, 马宁. 置管病人管理系统中 H5、TP5 关键技术的研究与实现[J]. 电脑编程技巧与维护, 2020, (04): 86-90.

[13] 刘晶. 高校固定资产管理系统的设计与实现[D]. 大连: 大连理工大学, 2018.

[14] Seth James Nielson, Christopher K. Monson. Asymmetric Encryption: Public/Private Keys[M]. Berkeley: Apress, 2019.

[15] Stepan Sivkov, Leonid Novikov, Galina Romanova, Anastasia Romanova, Denis Vaganov, Marat Valitov, Sergey Vasiliev. The algorithm development for operation of a computer vision system via the OpenCV library[J]. Procedia Computer Science, 2020, (10): 169-169.

[16] Guerrero-Sanchez Alma E, Rivas-Araiza Edgar A, Gonzalez-Cordoba Jose Luis, Toledano-Ayala Manuel, Takacs Andras. Blockchain Mechanism and Symmetric Encryption in A Wireless Sensor Network.[J]. Sensors (Basel, Switzerland), 2020, 20(10): 2798-2798.

[17] S. A. Malakh, V. V. Servakh. Maximization of Unit Present Profit in Inventory Management Systems[J]. Automation and Remote Control, 2020, 81(3): 843–852.

[18] 孙霓刚, 陈宣任, 朱浩然. 一种基于整数多项式环上的非对称全同态加密方案[J]. 现代电子技术, 2020, 43(05): 86-91.

[19] 毕红棋, 陈露. 基于混合算法的加密与解密的应用研究[J]. 现代信息科技, 2020, 4(03): 145-147, 150.

[20] 邵凯, 毛云龙. 二维码加密技术的研究[J]. 数字技术与应用, 2020, 38(02): 189-190.

在学研究成果

[1] 旷才英语-基于微信小程序的英语单词学习软件(V1.0)，软件著作权登记(登记号：2018SR651978)，第一著作权人。

[2] 旷才英语-基于微信小程序的英语单词学习软件，2018 年中国大学生计算机设计大赛二等奖，项目负责人。

致　谢

感谢我的导师邵中老师。在老师的悉心指导和严格要求下完成了本论文，导师的心血和汗水凝聚在本论文的各个方面：课题选择、方案论证到具体设计和实现。在四年的大学学习和生活期间，导师的精心教导让我更好的掌握和运用专业知识，并在论文中得以体现，不仅如此，日常的学习中也始终能够感受到导师无私的关怀。在此向邵中老师表示深深的感谢和崇高的敬意。

同时也感谢同学以及其他各位老师在思路想法上对我提供的帮助。感谢室友大学四年来的陪伴，共同走过这美好的大学生活。