

重庆市高校数据库应用程序设计大赛

作 品：菜鸟小邮——错取快递管理系统

参 赛 学 校：重庆邮电大学

参 赛 队 伍：菜鸟小邮

队员1学号与姓名：2020211370 龚南桥

队员2学号与姓名：2020211835梁 前

队员3学号与姓名：2020211442王 俊

指 导 老 师：吴桂军

2022年6月

目录

[错取快递管理系统 2](#_Toc15207)

[一、 背景介绍 2](#_Toc26859)

[1.1 应用场景背景介绍 2](#_Toc83)

[二、需求分析 3](#_Toc20146)

[2.1 系统需求分析 3](#_Toc2170)

[2.2 功能需求分析 4](#_Toc1123)

[2.2.1 快递驿站端功能分析 4](#_Toc3352)

[2.2.2 用户端功能分析 5](#_Toc21788)

[三、概念结构设计 7](#_Toc4184)

[3.1 系统总体框架 7](#_Toc14267)

[3.2 用户端和驿站端框架 8](#_Toc15739)

[四、 数据库设计 9](#_Toc25332)

[4.1 E-R图 9](#_Toc3854)

[4.2 模型设计 10](#_Toc13084)

[4.3 数据库设计与实施 11](#_Toc5135)

[4.3.1 数据库设计细节 11](#_Toc30545)

[4.3.2 创建数据库的SQL语句 14](#_Toc13035)

[五、 系统实现 16](#_Toc10473)

[5.1 技术栈 16](#_Toc444)

[5.2 各功能界面 18](#_Toc26444)

## 错取快递管理系统

## 背景介绍

### 应用场景背景介绍

得益于我国庞大、快速、便捷的物流网络和健全、强大的互联网体系，线上购物这一新兴的消费方式正在深刻改变着我们每个人的生活。各种快递代收点如雨后春笋般相继出现，阿里巴巴旗下的“菜鸟驿站”是快递代收点中当之无愧的领头羊、巨无霸。“菜鸟驿站”的各项功能已经非常完善，但是面对快递错取这一场景仍然存在不足。

“菜鸟驿站”采取的是“自主取件，扫码出库”这一快递出库体系，在出库过程中缺乏人为检查和监督，用户主观性极大，所以可能会出现取错快递的情况，但是面对这一情况“菜鸟驿站”并未有相应的应对措施。这就造成了被取错快递的用户并不知道自己的快递被取错了；错取了别人快递的用户是否归还快递、何时何地归还快递全凭用户自己的道德水平和主观意志。

针对这一场景我们设计了“CQUPT错取快递管理系统”，该系统旨在帮助用户处理错取快递这一情形，让被取错快递的用于即时得知自己的快递被取错了，何时被取错了以及被谁取错了；提醒错取快递的用户取了别人的快递，并督促用户即时将快递归还至快递驿站。

除此之外,在新冠疫情的时代背景下，快递也具有传播新冠的潜在风险。传统的取件方式为人工核对快递信息是否与取件人的信息一致，这样会增加取件人在快递驿站逗留的时间，增大取件时感染新冠的风险。基于上述考虑，我们还为该系统设计了人脸识别取件功能，这样有助于减少取件人在快递驿站的逗留时间，可以减少感染新冠的风险。

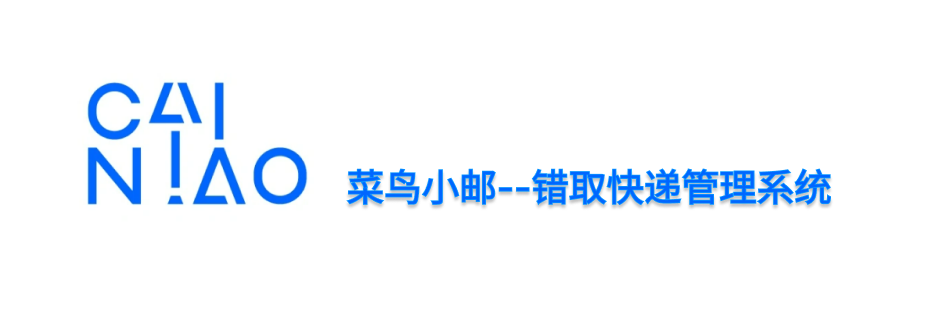


图1 项目logo

## 二、需求分析

### 2.1 系统需求分析

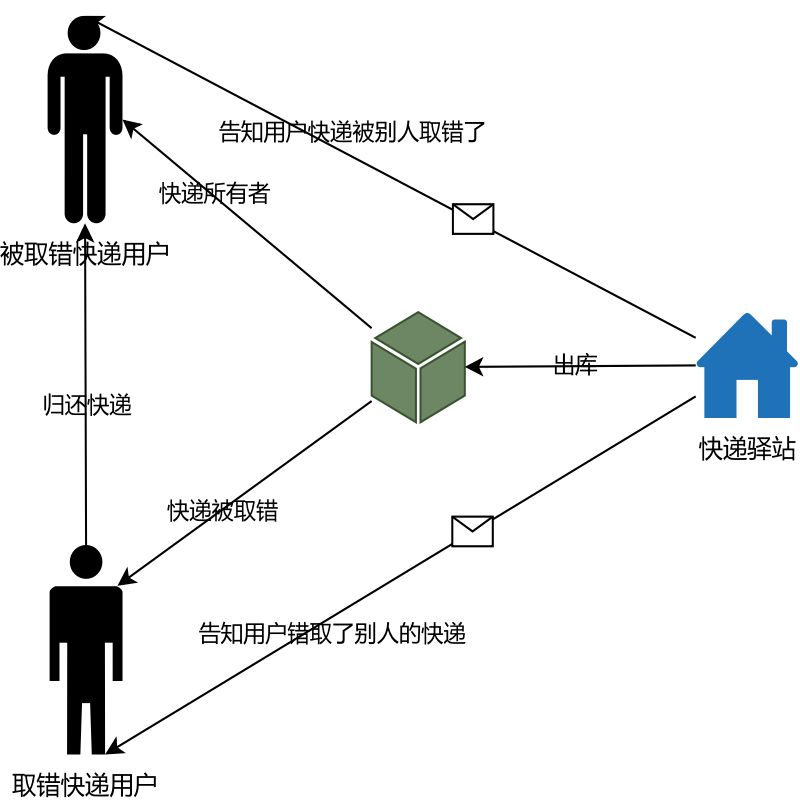


图2 需求分析图

该系统包含两个主体：快递驿站和用户，当快递被非快递所有者取件出库后快递驿站需要通知快递所有者他的快递被别人去错了，需要让该用户知道错取了他快递的用户的部分个人信息，以便在快递长时间未归还后该用户私下与错取快递的用户联系；快递驿站还需要通知错取了别人快递的用户，他取了别人的快递，需要提醒该用户将快递即时归还给快递所有者。

### 2.2 功能需求分析

#### 2.2.1 快递驿站端功能分析

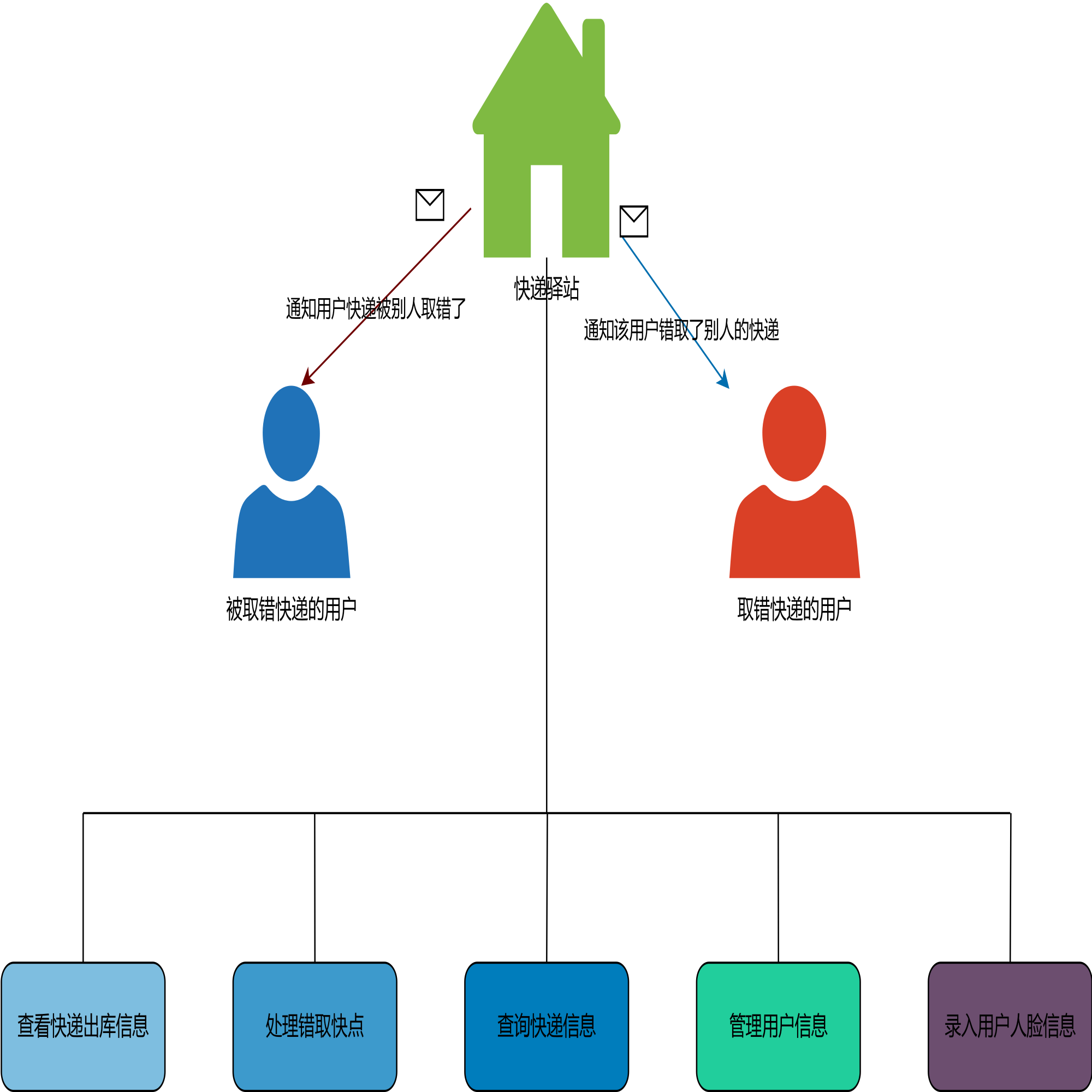


图3 快递驿站端功能分析

由于我们聚焦的是错取快递这一情景，所以快递出入库、快递在驿站中的位置分配、通知用户取件和扫码取件功能不在我们的考虑范围之内，所以这部分功能我们不会涉及。

快递驿站端主要是对被取错的快递进行管理。当检测到某一个快递被取错后快递驿站端需要通知被取错快递的用户这一情况，并将错取者的电话信息、姓名、所住楼栋等信息发送给被取错快递的用户，与此同时快递驿站也需要提醒取错快递的用户他错取了别人的快递并通知该用户尽快将快递归还到快递驿站。

被非快递所有者所取的快递并不一定是该用户取错了，也有可能是快递所有者委托该用户代取的快递。面对这种情况我们选择让快递原本的所有者来决定该快递是否是被取错了，如果用户确定该快递是被取错了，那么快递驿站将会即时收到消息并将取件者信息发送给快递原本的所有者，以便快递所有者联系错取快递者。

除此之外，快递驿站端还需要有人脸识别功能，方便用户使用人脸取件。

#### 2.2.2 用户端功能分析

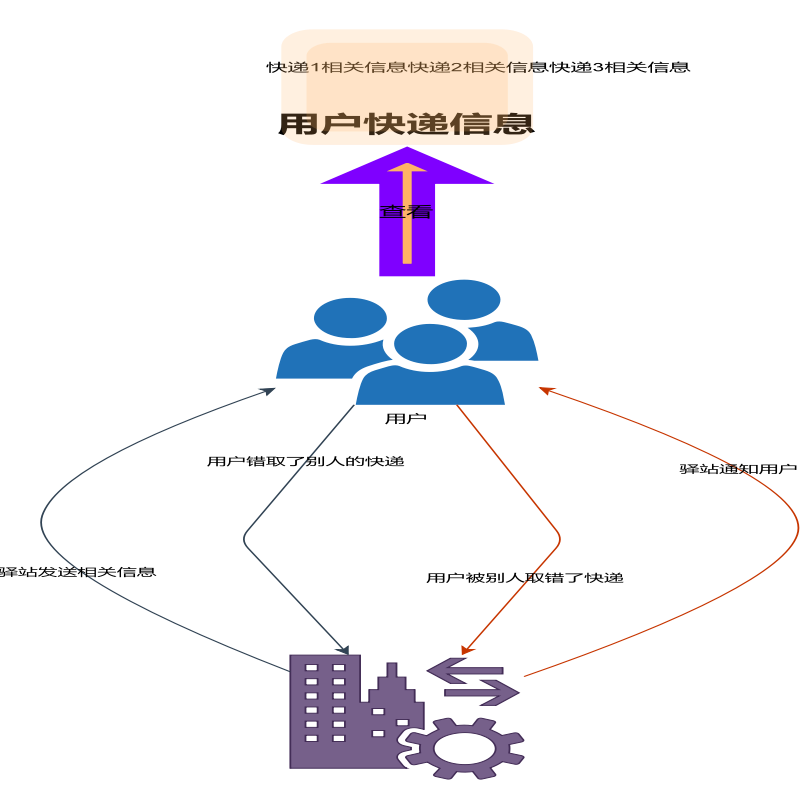


图4 用户端功能分析

在用户端我们需要实现让用户查看自己的快递情况。需要能够提醒用户去快递驿站取自己未取的快递；如果用户错取了别人的快递（帮别的用户取的快递除外），需要提醒用户错取了别人的快递并让用户即时将快递归还到快递驿站；如果用户的快递被别人取错（未委托 别的用户代取快递），需要让用户知道是谁取错了他的快递，如果快递未被及时归还到快递驿站，则需要让用户知道错取快递者的联系方式以及所住楼栋，以便用户催促快递的归还。

## 三、概念结构设计

### 3.1 系统总体框架

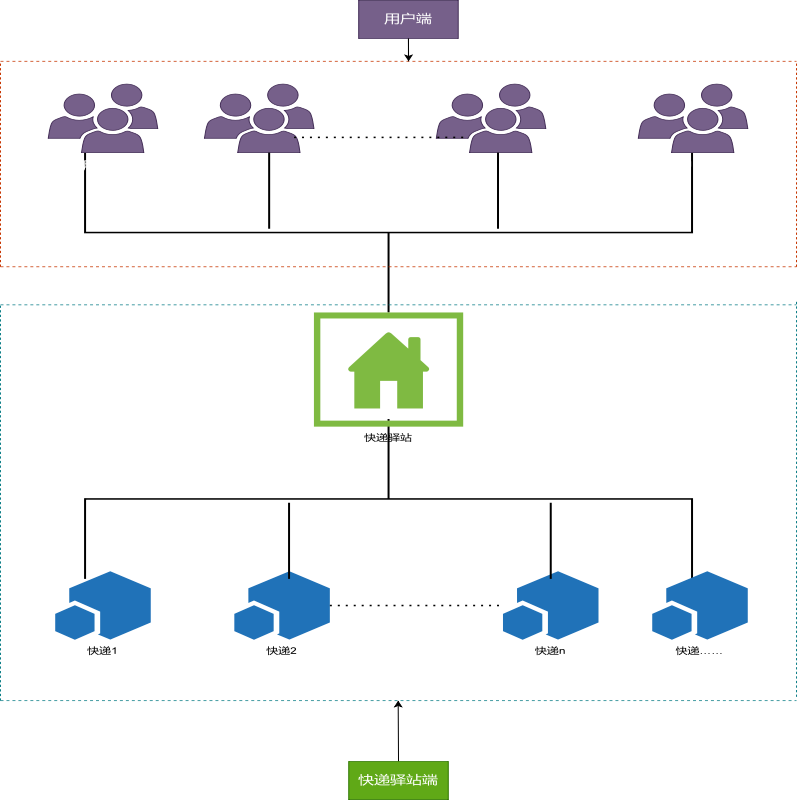


图5 系统总体框架

### 3.2 用户端和驿站端框架

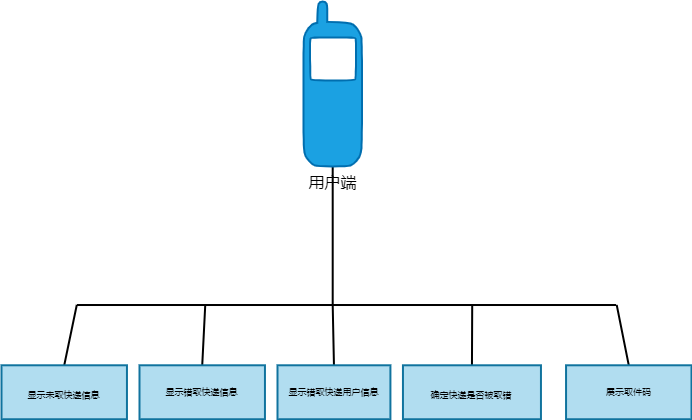
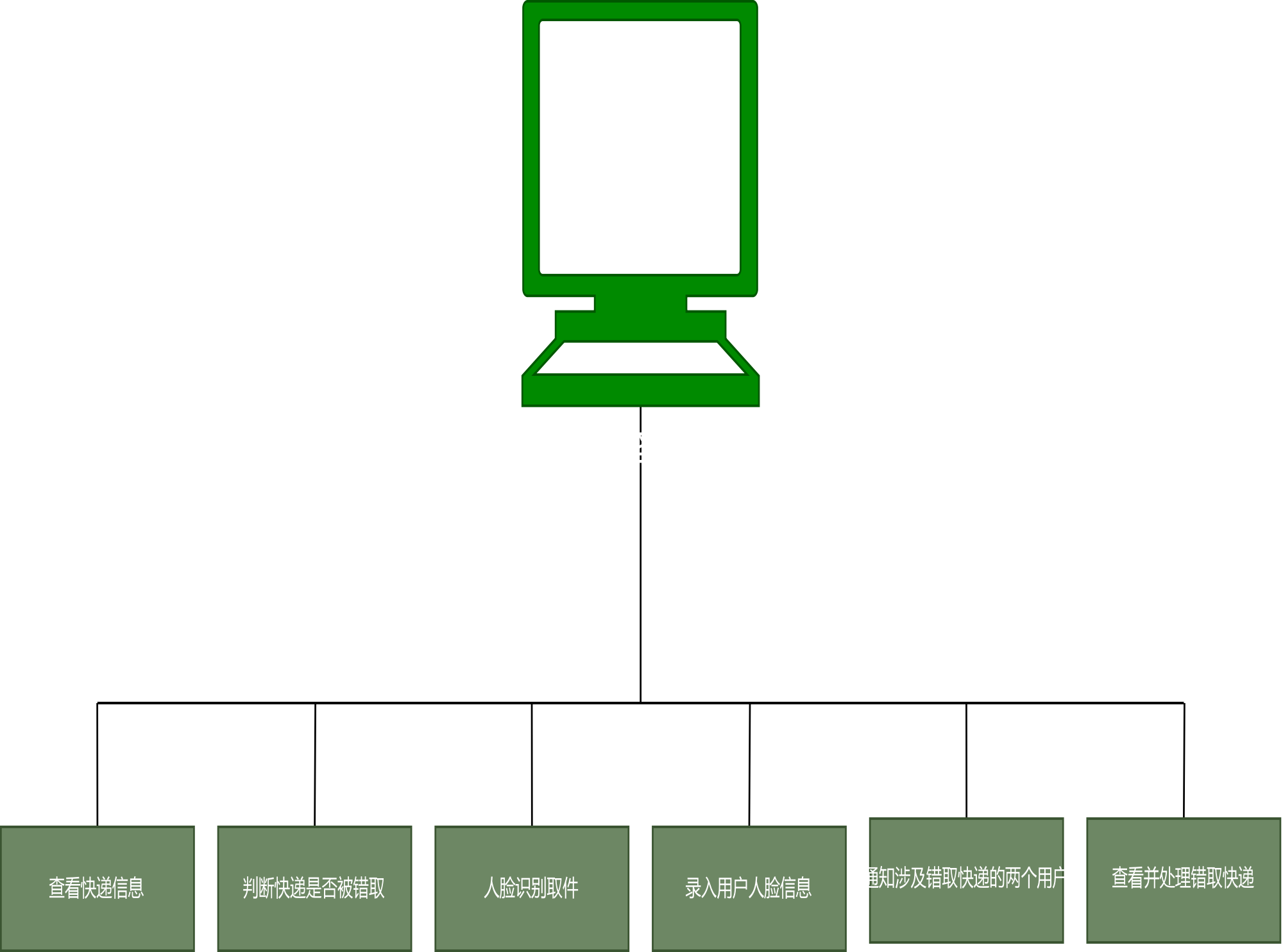


图6 用户端框架



驿站端

图7 驿站端框架

## 数据库设计

### 4.1 E-R图

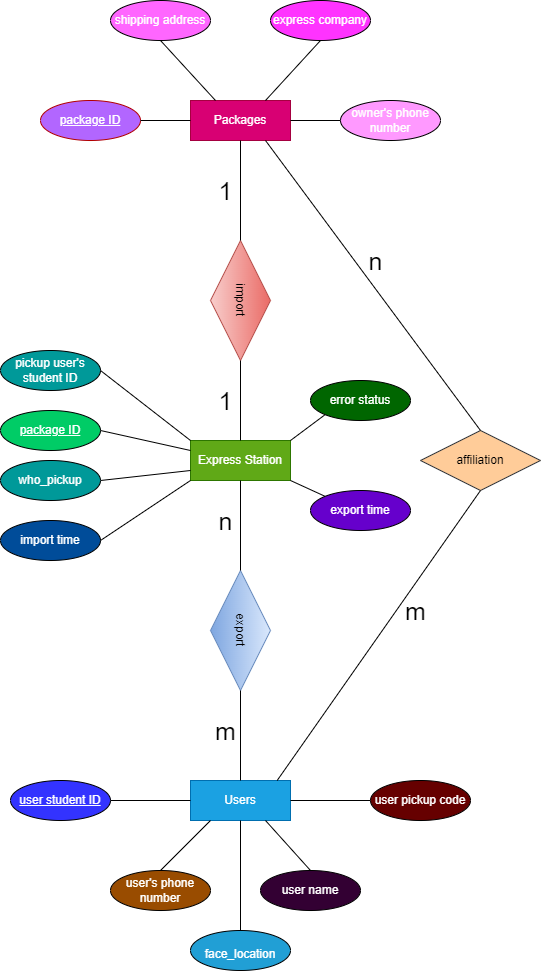


图8 E-R图

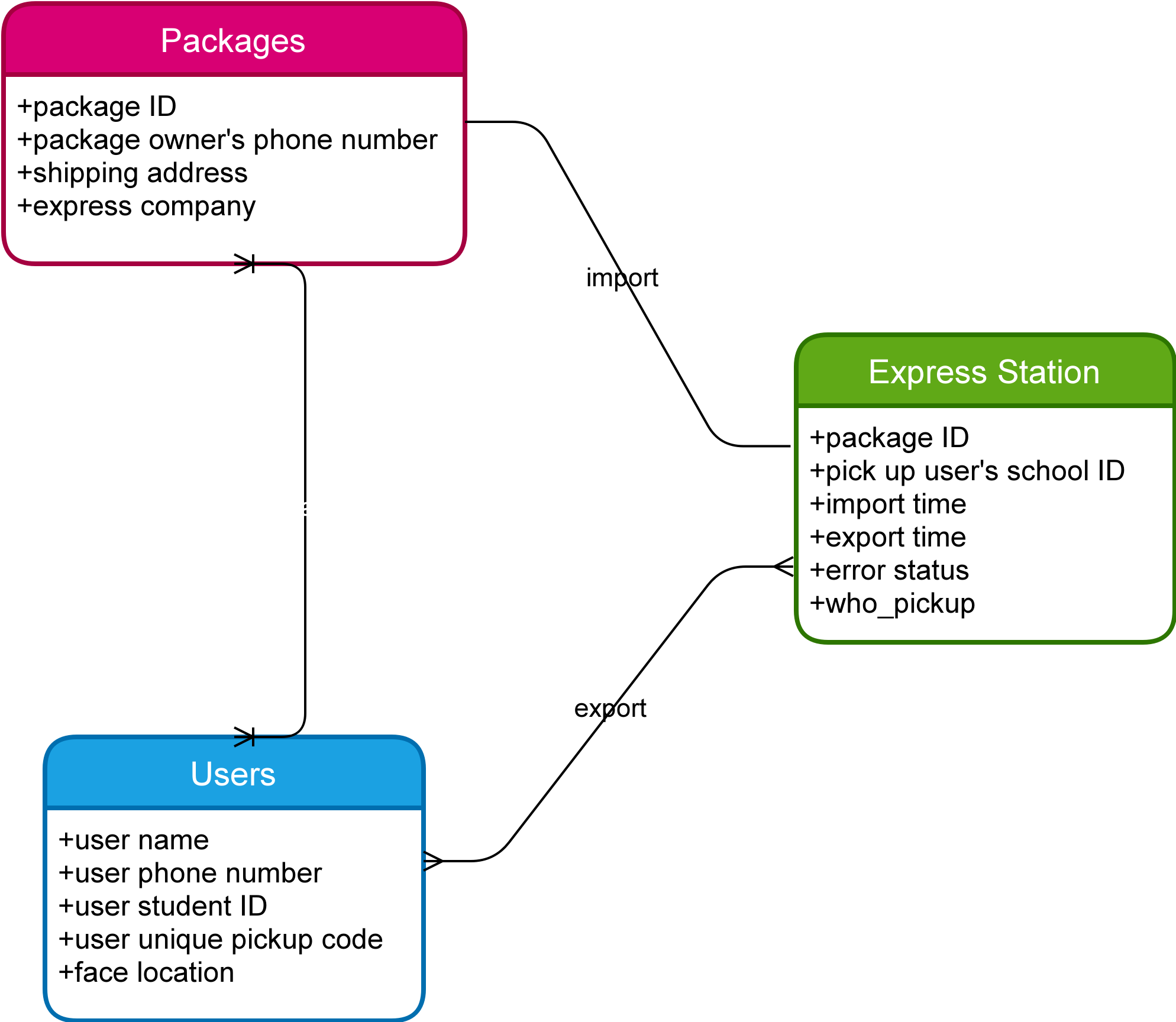


图9 UML图

### 4.2 模型设计

由前面的需求分析以及E-R图设计，我们大致需要设计三张表，一张Packages表，用以记录快递的信息；一张Express Station表，用于记录快递的出入库相关信息；一张Users表，用于记录用户的个人信息。

CDM图如下所示（该图是使用workbench所画，在workbench中表间的关系不直接显示）：

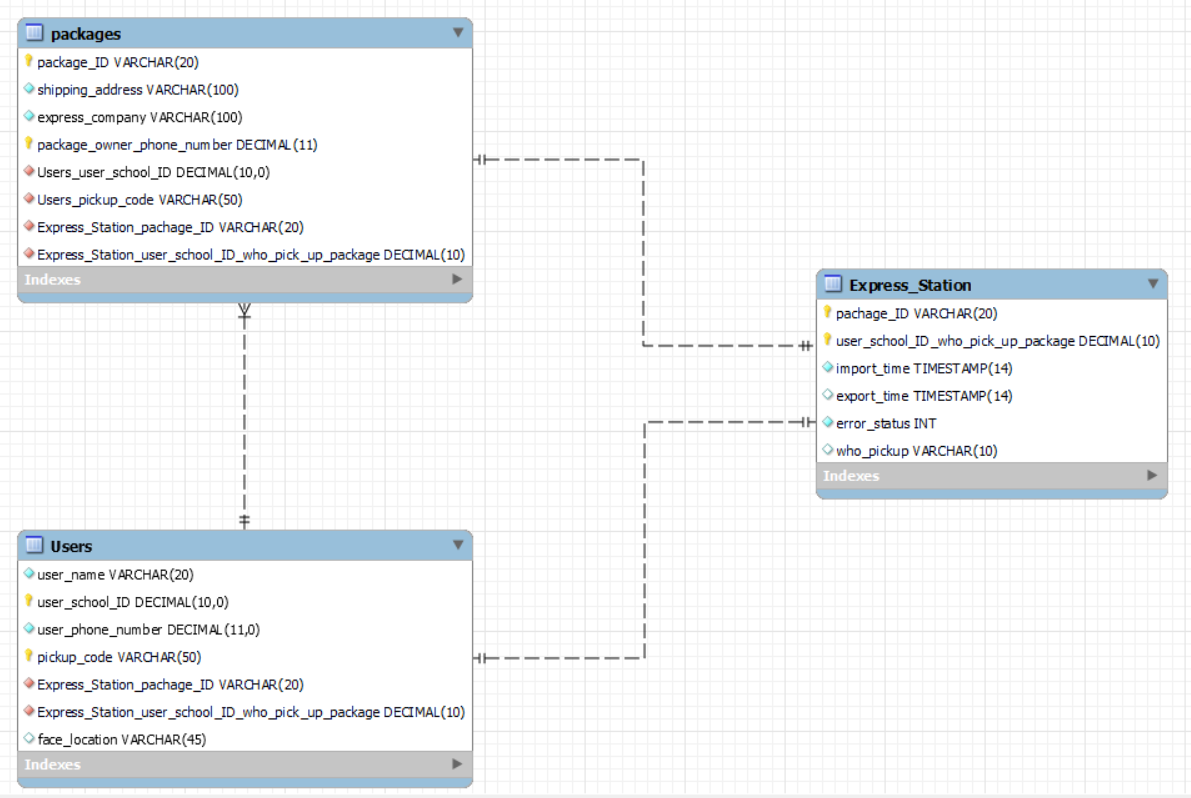


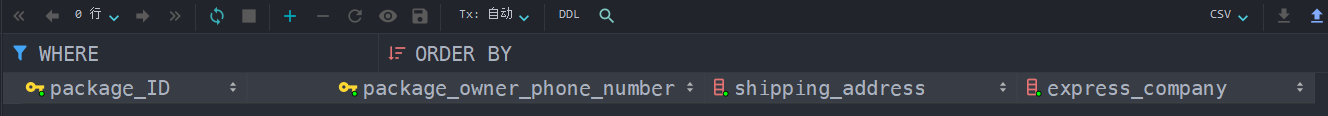
图10 CDM图

### 4.3 数据库设计与实施

#### 4.3.1 数据库设计细节

我们创建一个数据库——CQUPTEMS\_DB，该数据库用于存储所有的数据。数据库中由三张表，就是前面提到的三张表：Packages、Express Station和Users。

Packages表在Datagrip中 如下图所示：



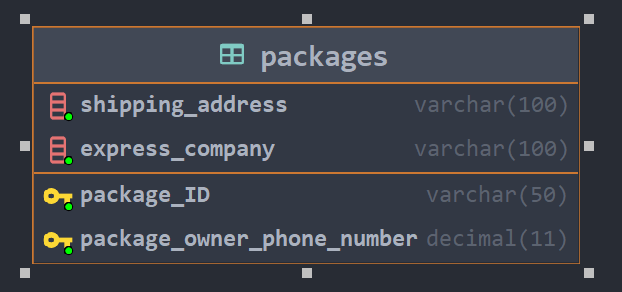
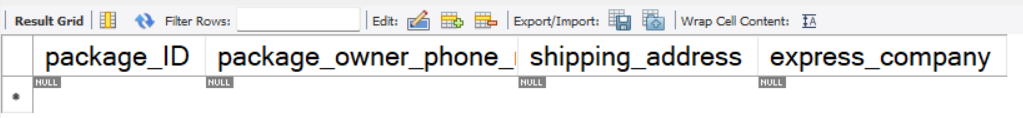


图11 Datagrip中的Packages表

在workbench中如下图所示：



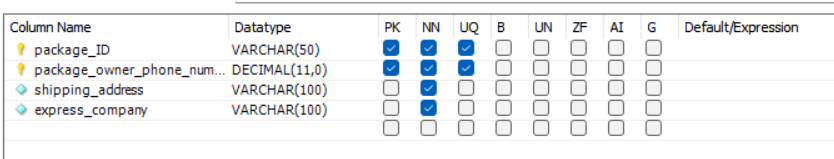


图12 workbench中的Packages表

Express Station表在Datagrip中如下图所示：

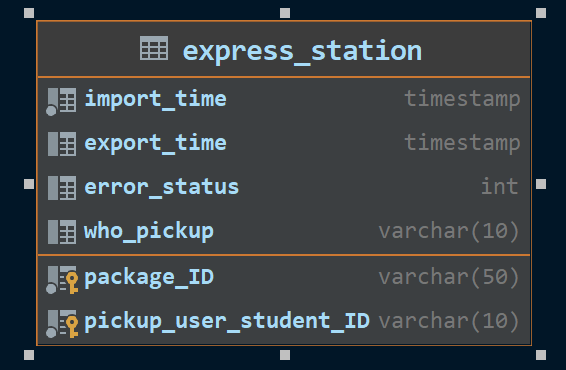
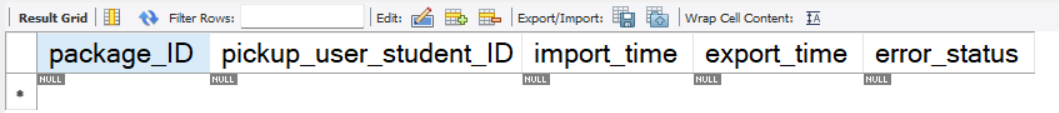


图13 Datagrip中的ExpressStation表

在workbench中如下图所示：



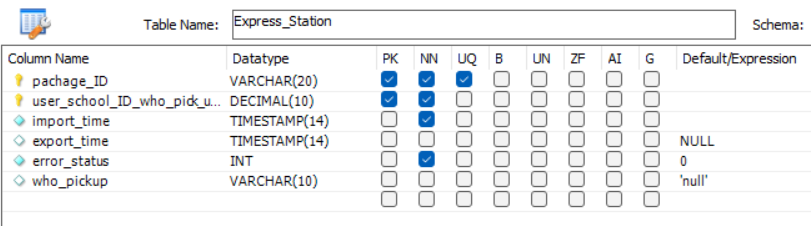


图14 workbench中的ExpressStation表

Users表在Datagrip中如下图所示：

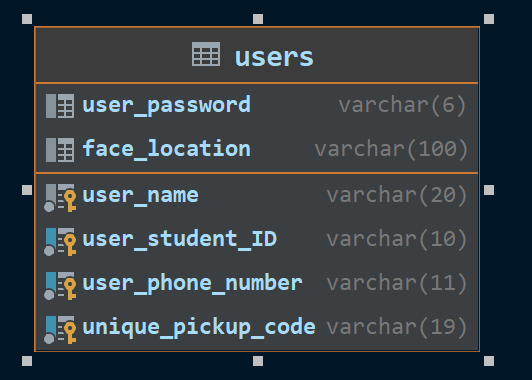
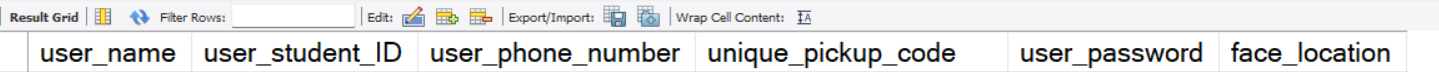


图15 Datagrip中的Users表

在workbench中如下图所示：



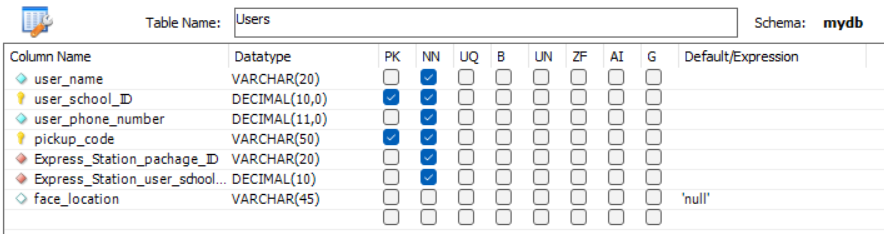


图16 workbench中的Users表

#### 4.3.2 创建数据库的SQL语句

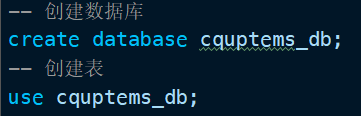


图17 创建数据库的语句

首先创建数据库；

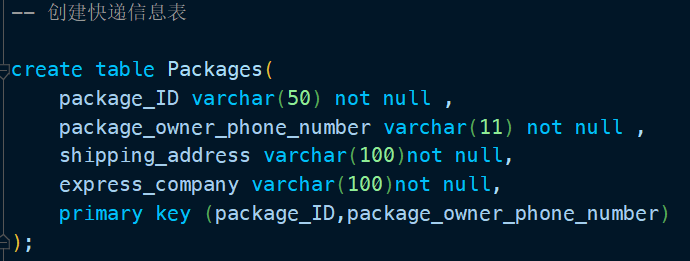


图18 创建Packages表的MySQL语句

然后创建快递信息表；

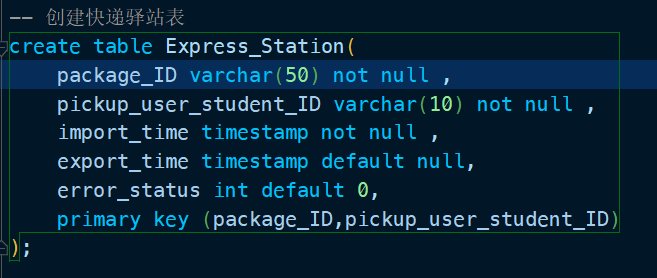


图19 创建Express Station表的MySQL语句

然后创建快递驿站表；

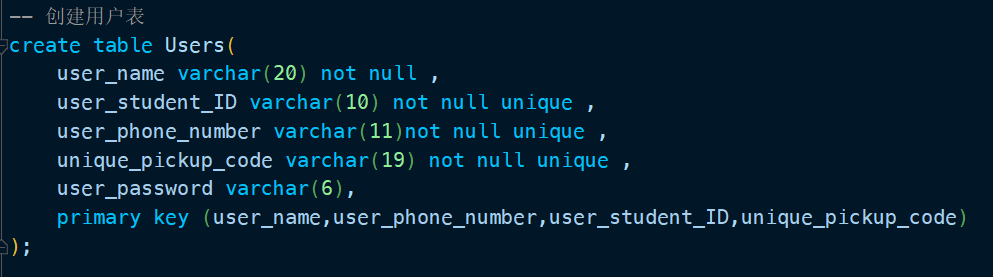


图20 创建Users表的MySQL语句

创建用户信息表；

值得注意的是，上述只是初步创建了表，后续对某些表增加了属性，如下图所示：

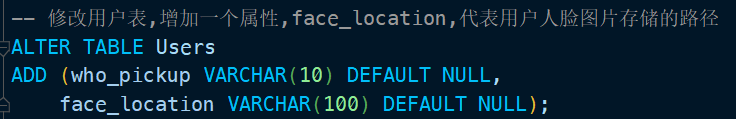


图21 向Users表中增加属性的MySQL语句

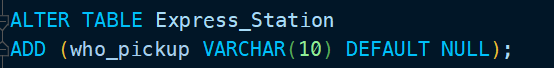


图22 向Express Station表中增加属性的语句

除了创建上述表之外，我们还在表Packages上创建了一个触发器，如下图所示：

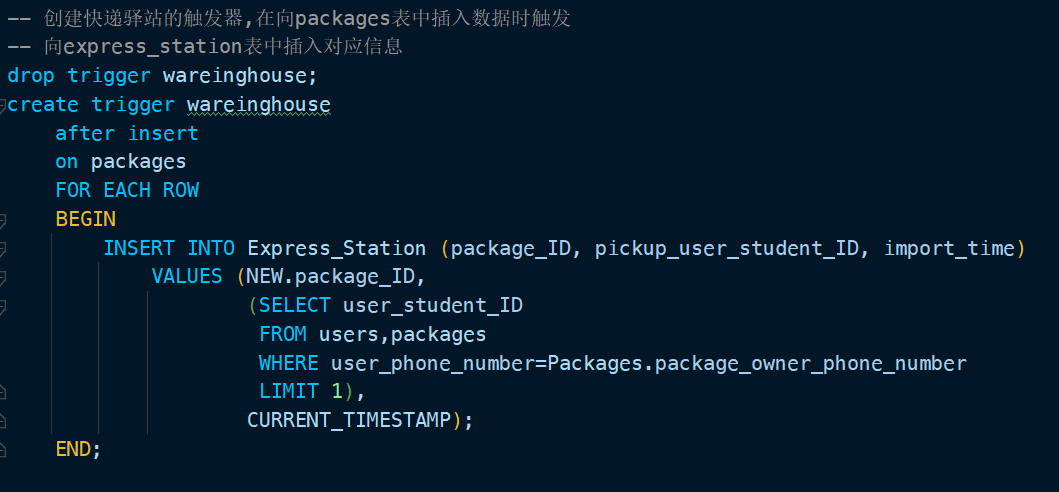


图23 在Packages表上创建触发器的MySQL语句

该触发器用于在向Packages表中插入数据后向Express station表中插入相应的信息，用于模拟一个快递入库的过程。

## 系统实现

### 5.1 技术栈

整个系统的实现我们选择的是本地GUI的形式，主要有python进行开发。前端设计方面，我们选择使用Figma来实现，后端我们采用python的tkinter来实现。由于Figma到tkinter代码不能直接转化，所以中间需要用到一个python开源工具tkinter-designer，这个工具能将设计好的Figma前端页面效果图转化为相对应的tkinter代码。

数据库方面，我们使用的是MySQL。

人脸识别方面，底层我们使用的是C++的第三方开源库dlib，它集成了许多业界领先的机器学习算法，我们具体使用的是实时人脸识别方面的HOG（histogram of oriented gradient）算法和ERT（ensemble of regression trees）级联回归算法，即基于梯度提高学习的回归树方法；在其上层，我们使用的是将dlib的各接口封装好的python第三方开源库face\_recognition；在图像的处理方面，我们使用的是OpenCV。

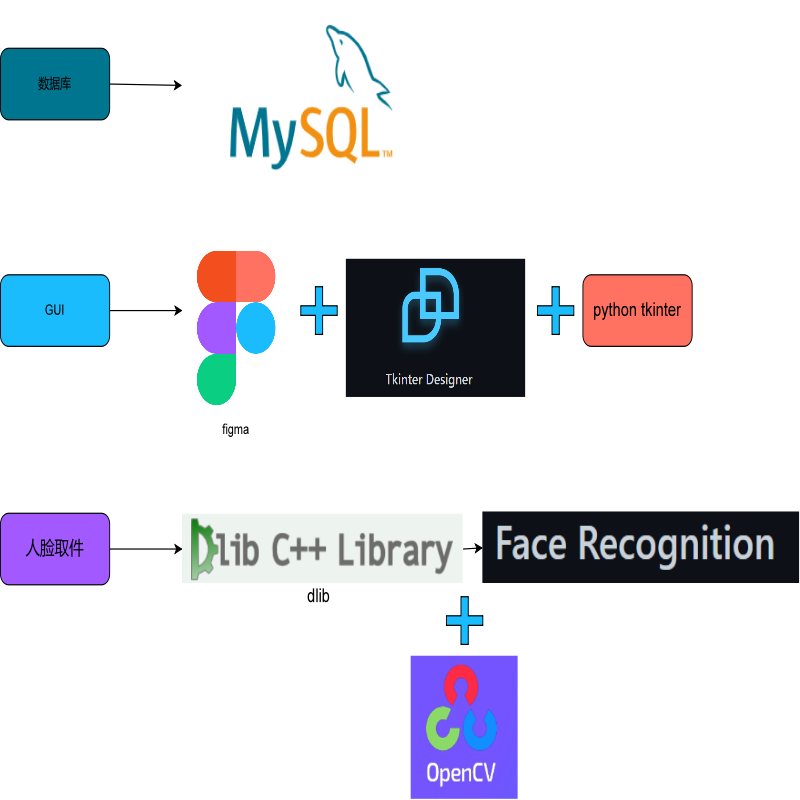


图24 技术栈

### 5.2 各功能界面

#### 5.2.1 登录和注册界面



图25 注册界面

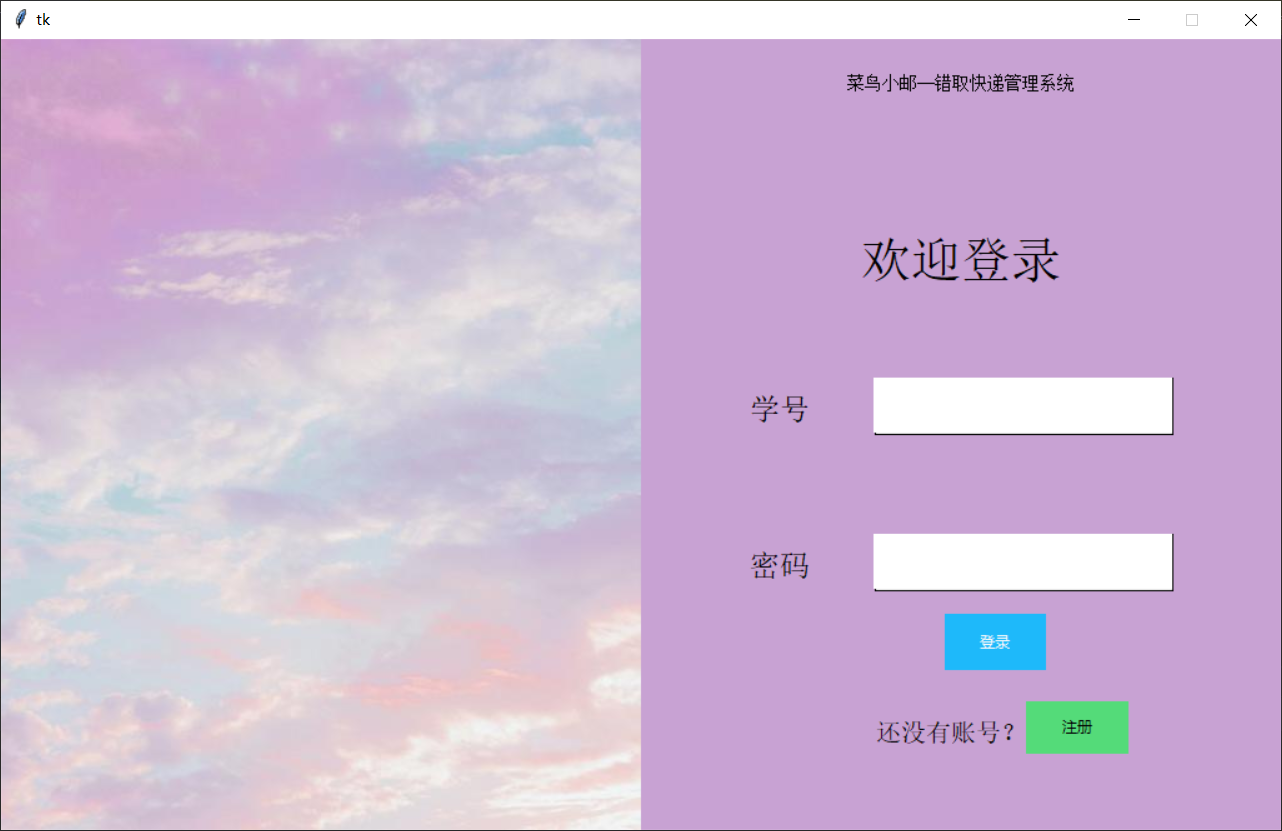


图26 登录界面

在注册界面中，用户需要输入姓名、学号、电话和密码等信息才能完成注册。如果用户已有账号可以可以选择点击登录按钮，界面就会跳转到登录界面。

在登录界面中，用户则需要输入学号和密码才能登录。

#### 5.2.2 用户界面



图27 用户界面

在该界面中，用户能够查看自己的相关快递信息，例如：未取快递信息、已取快递信息、错取快递信息和自己的取件码。该界面主要是面向用户，所以只应该有相关的查询功能。

#### 5.2.3 快递驿站界面

图28 快递驿站界面

该界面是快递驿站界面，也是管理员界面，有很高的权限，能够实现增删改查等功能。该界面能看到所有的快递信息，如已出库快递、未出库快递和错取快递信息等。

#### 5.2.4 人脸识别取件界面

图29 人脸识别取件界面

该界面主要是显示从摄像头获取的视频流信息，也就是将摄像头获取的图像显示到界面上。点击人脸识别按钮后就可以开始进行人脸识别。

人脸识别实例：

图30 人脸识别取件实例

## 附录

### 6.1 开发工具

开发语言：python，MySQL

开发工具：Visual Studio Code，Pycharm，Datagrip，Workbench，Anaconda

技术栈：tkinter，tkinter-designer，figma，dlib，face-recognition，opencv，mysql

### 6.2 依赖项

由于该项目使用了很多第三方开源库，有一些第三方开源库又依赖其他的第三方库，而且有一些是C++库，需要自己在电脑上编译安装，所以本项目的依赖非常多，所涉及的python第三方库高达近400项。

所有的依赖项都已经在项目的requirements.txt文件中说明，要安装这些依赖项只需在项目目录下的控制台或powershell中执行语句“pip install -r requirements.txt”即可。

部分依赖项如下图所示：

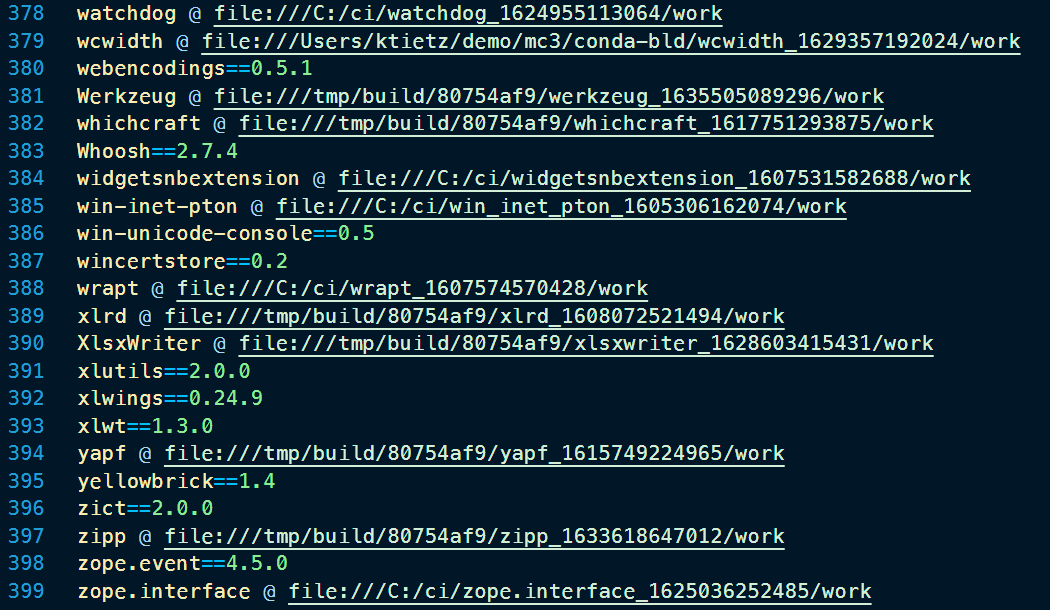


图31 requirements.txt中的部分依赖项

### 6.3 开源许可

该项目已经在GitHub上开源，开源证书为Apache License，项目地址为：<https://github.com/NaNBridge/CQUPT_Express_Management_System.git>

部分开源证书信息如下图所示：

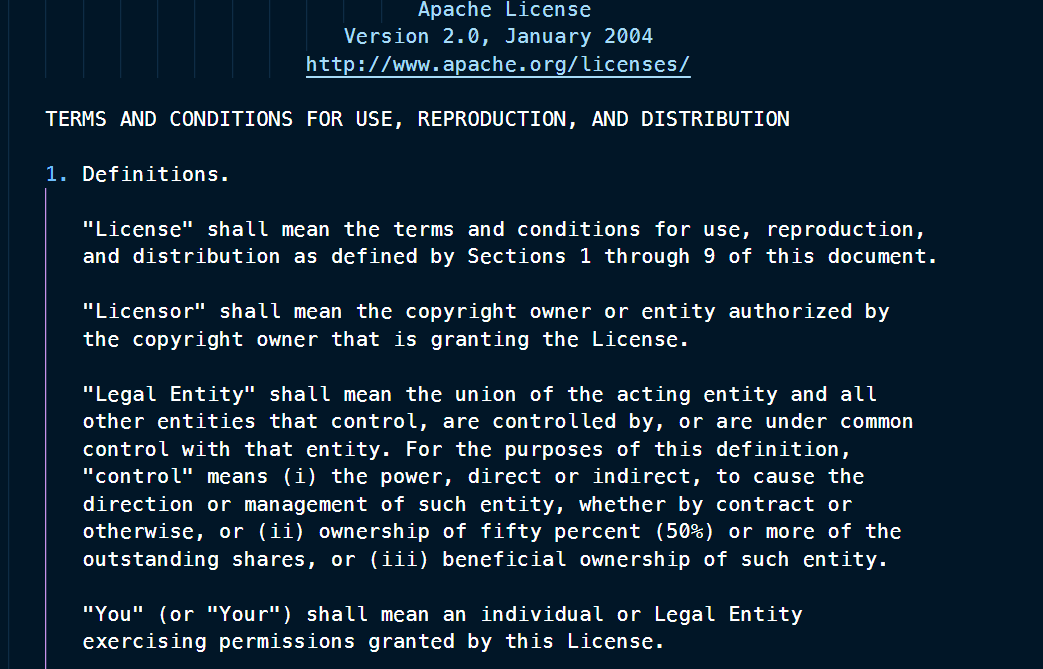


图32 部分开源证书信息