**<基于PKI的二维码安全认证系统>**

**用户手册**

（V1.0）

**目录**

[基于PKI的二维码安全认证系统户手册 1](#_Toc472862029)

[1 产品介绍 1](#_Toc472862030)

[1.1 产品简介 1](#_Toc472862031)

[1.2 产品功能 1](#_Toc472862032)

[2 基本概念 2](#_Toc472862033)

[3 产品及其他组件介绍 3](#_Toc472862034)

[3.1 产品 3](#_Toc472862035)

[4 操作指南 4](#_Toc472862037)

[4.1 快速使用指南 4](#_Toc472862038)

[4.2 编译与开发环境 4](#_Toc472862039)

[4.2.1 安卓工程的构建 5](#_Toc480297121)

[4.2.2 Web前端工程的构建 5](#_Toc480297121)

[4.2.3 Web后台工程的构建和部署 5](#_Toc480297121)

[附录A 技术规格 6](#_Toc472862041)

# 产品介绍

## 产品简介

简要介绍整个产品的特点、应用场景等。

介绍产品符合的国密规范，接口等。

本产品是安装在Android系统上与Web端配合使用的二维码安全认证系统，应用于用户安全的扫码登录和电子合同签署认证场景。基于PKI密码体系，采用国际标准的RSA加密技术进行签名和认证，具有较好的安全性。

## 产品功能

介绍整个产品的功能特点，主要突出关键技术。

本产品在二维码的使用背景下，结合PKI技术进行身份认证，可以实现手机免密在Web端扫描二维码登录，利用Web后台存储包含公钥的用户证书和Android端私钥文件进行用户身份认证。同时，利用手机中的私钥文件还可实现对电子合同的签名认证功能。软件操作简单、易用，符合人们日常生活中的使用习惯，避免复杂的操作流程。界面设计风格统一，便于使用者快速掌握系统操作方法；用户界面简单明了，符合业务办理流程，具有人性化特点。

# 基本概念

介绍本产品的应用术语，如密码卡介绍设备编号、设备型号、密钥索引等不被客户理解的专业术语。

|  |  |
| --- | --- |
| 专业术语 | 术语简介 |
| 口令 | 用于加密Android本地的用户私钥文件，防止被人恶意利用 |
|  |  |
|  |  |

# 产品及其他组件介绍

## 产品

以图片结合文字方式介绍产品的各个组成部分，各部分安装前后对比示意图等。

**Android客户端**

1. 初始界面

点击Android端应用图标之后进入初始界面，界面显示<注册>和<登录>两个按钮

<注册>

新用户进行点击此按钮进行用户注册

<登录>

已注册用户通过此按钮进行登录

2、注册界面

点击<注册>按钮之后进入注册界面，根据提示填写相关信息进行注册

真实姓名 – 产品使用者的真实姓名

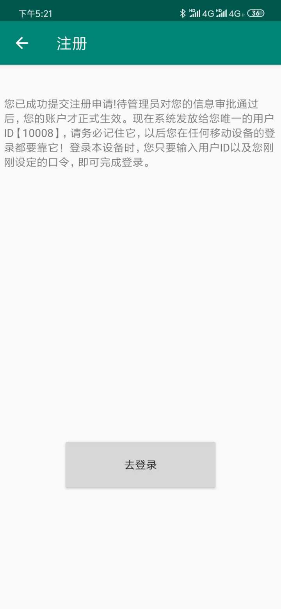
身份证号 – 产品使用者的身份证号

设置口令 – 口令用于保护本地存储用户信息

数据

<申请注册>

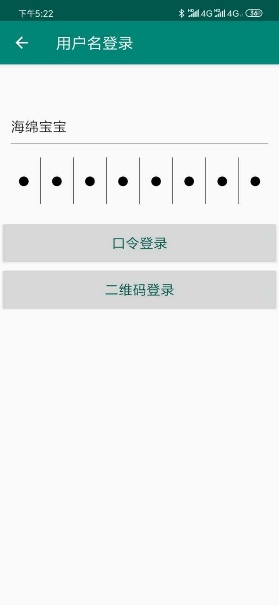
填写信息核验通过之后注册成功即可获得相应的用户ID

3、注册成功

注册通过审批之后跳转至这一界面，显示注册已成功，可选择<去登录>

<去登录>

注册成功使用刚申请的账号和口令进行登录

4、登录界面

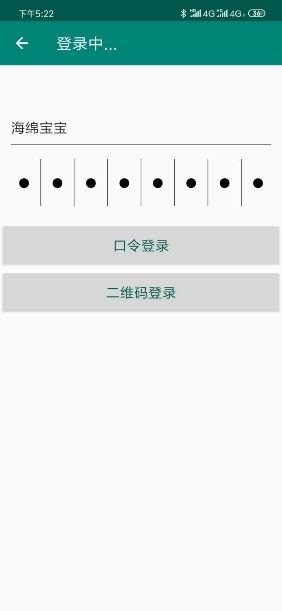
点击<登录>按钮之后进入登录界面，具有<口令登录>和<二维码登录>两种登录方式

<口令登录>

在输入框中输入之前注册通过的 ’用户账号’ 与 ’口令’ 进行登录，登录成功之后跳转用户中心

<二维码登录>

用于首次登录该设备，点击后使用已登录设备进行扫码登录

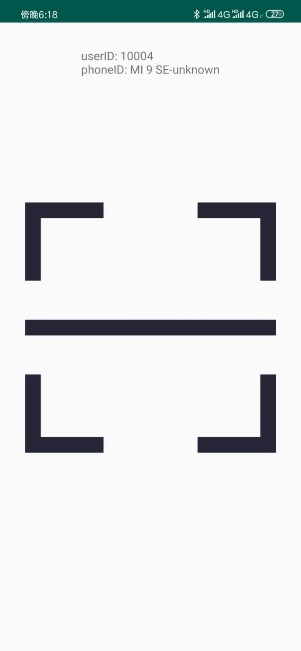


5、口令登录

输入正确的”账户名”和”口令”，点击<口令登录>按钮进行登录

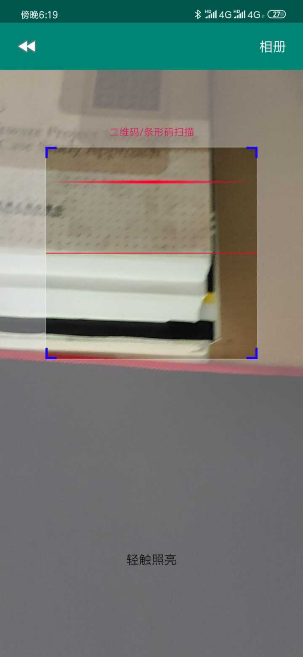
 5、二维码登录

点击<二维码登录>之后显示可供扫描的二维码，使用已登录的设备扫描二维码授权登录



6、用户中心界面

<口令登录>及<二维码扫码>登录之后进入用户中心界面，点击中心扫码图标可使用扫一扫功能扫描其他新设备授权登录，或者对电子合同的扫码签名认证

7、扫码界面

点击中心扫码图标后开始使用扫一扫功能，将摄像头对准二维码进行扫码操作



8、Web授权登录界面

扫描Web页面提供的电子合同签名二维码后可选择<确认授权>或者<取消>

<确认授权>

对Web界面登录进行授权

<取消>

取消授权

9、电子合同签名界面

扫描Web页面提供的电子合同签名二维码后可选择<签名>或者<取消>

<签名>

完成对电子合同的签名

<取消>

取消这一次合同的签名授权

**Web客户端**

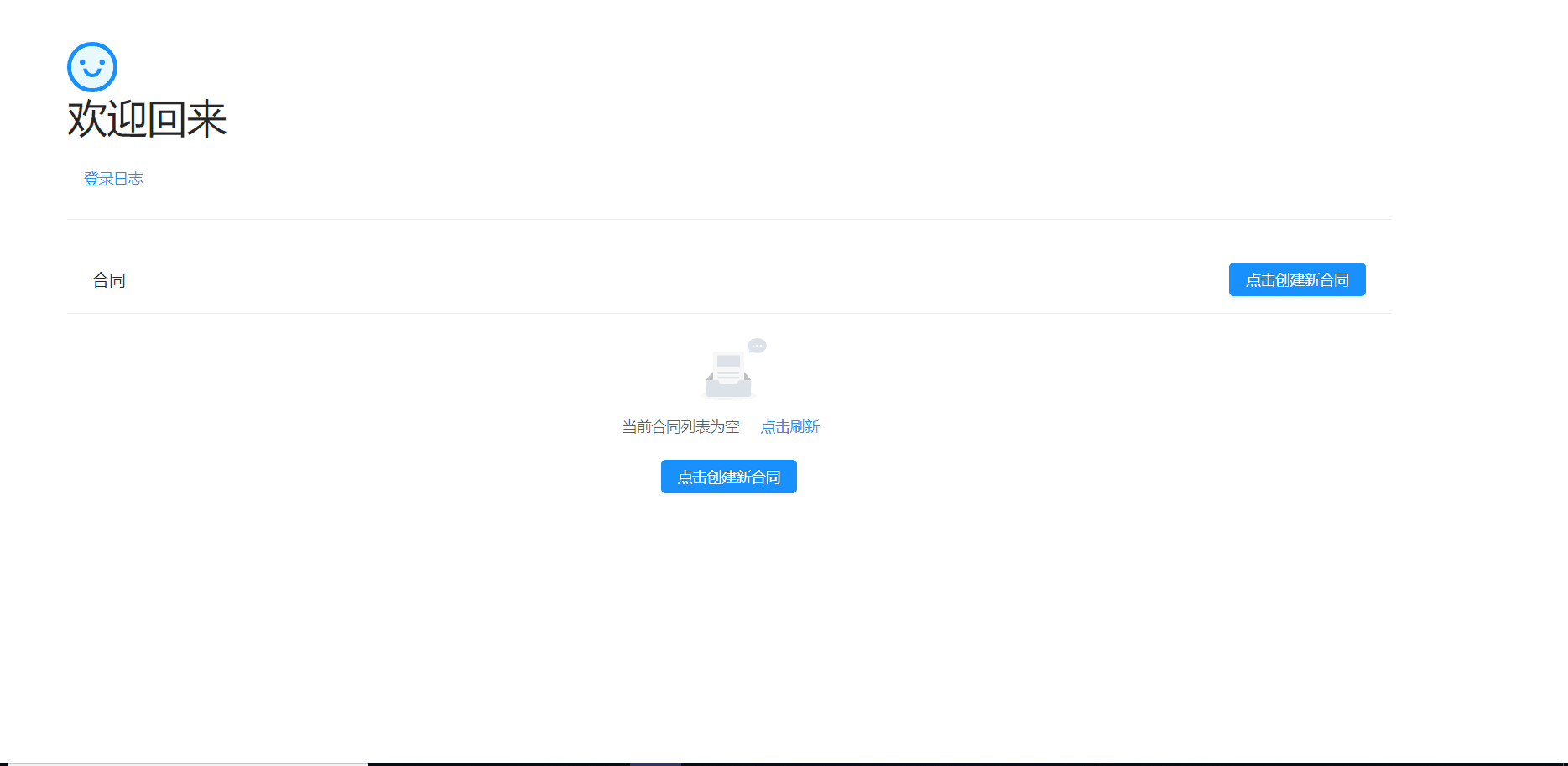
1. 二维码扫码登录页面

Web端登录需使用手机App扫码进行登录



2、Web主页面

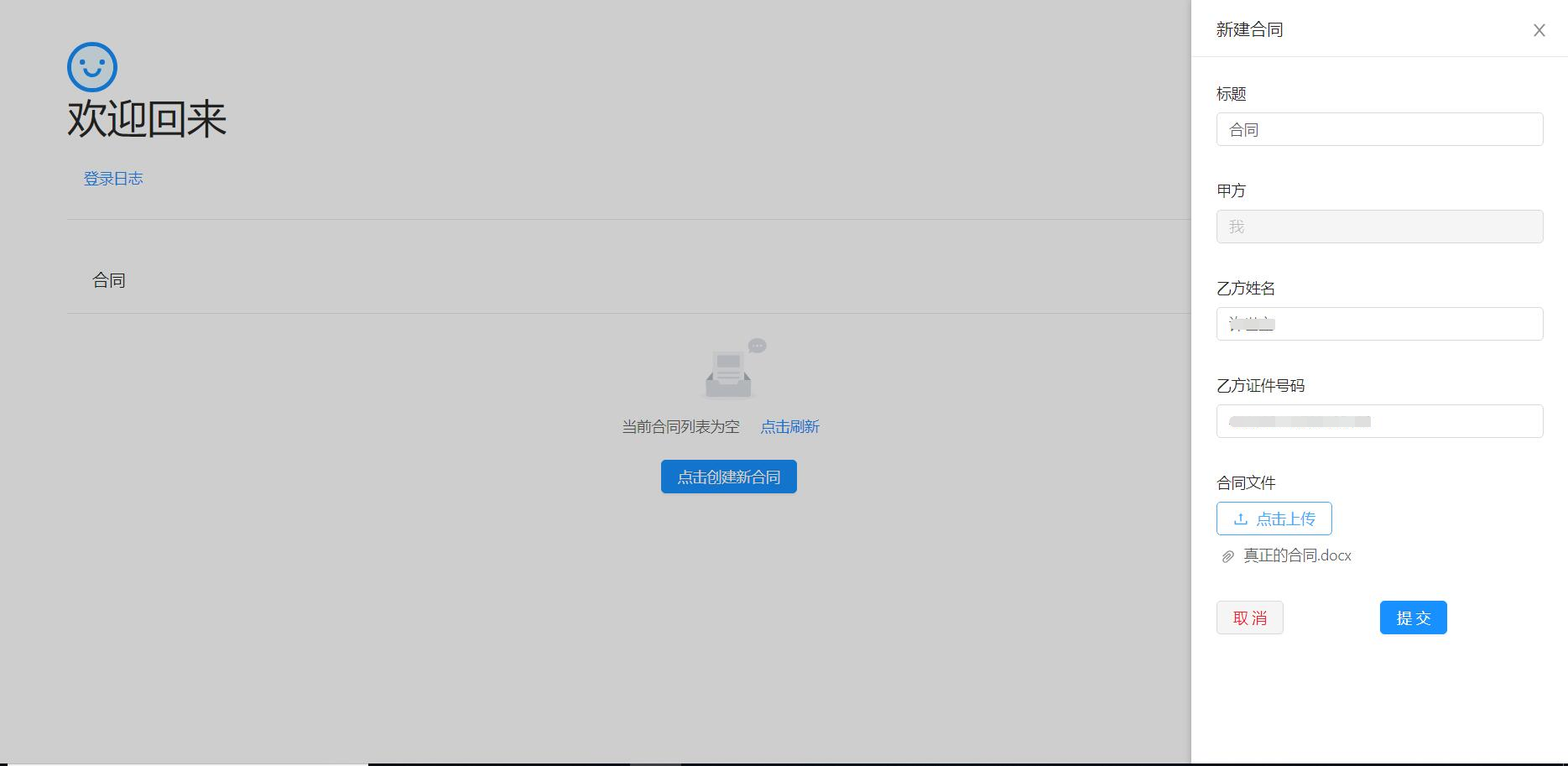
登录成功之后跳转至这一界面，可查看<登录日志>以及<点击创建新合同>进行新合同的创建





3、创建新合同

甲方填写相关有效信息进行新合同的创建，并上传相应的合同文件，最后点击提交即可完成创建过程，取消则不创建



4、合同创建成功

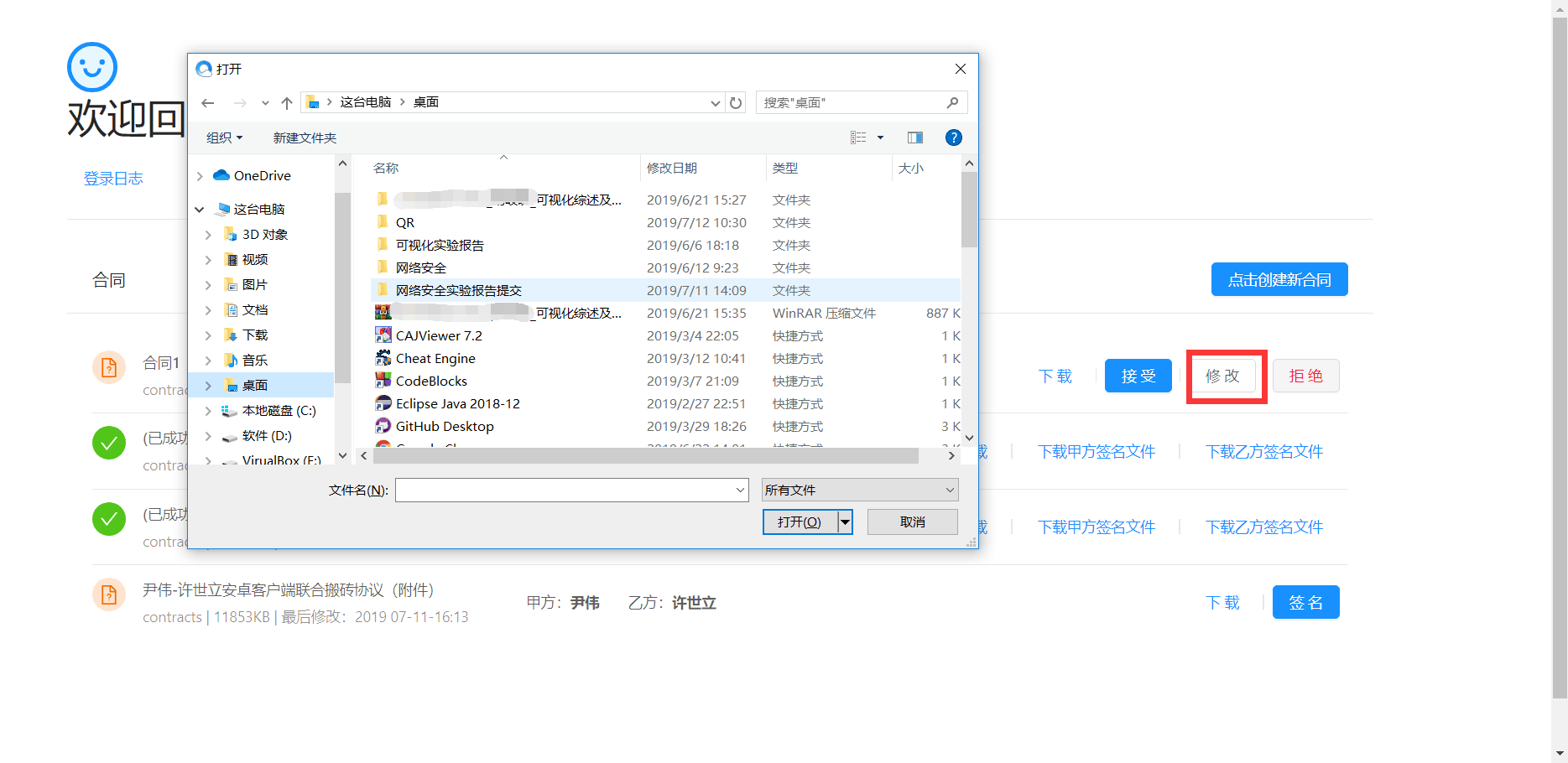
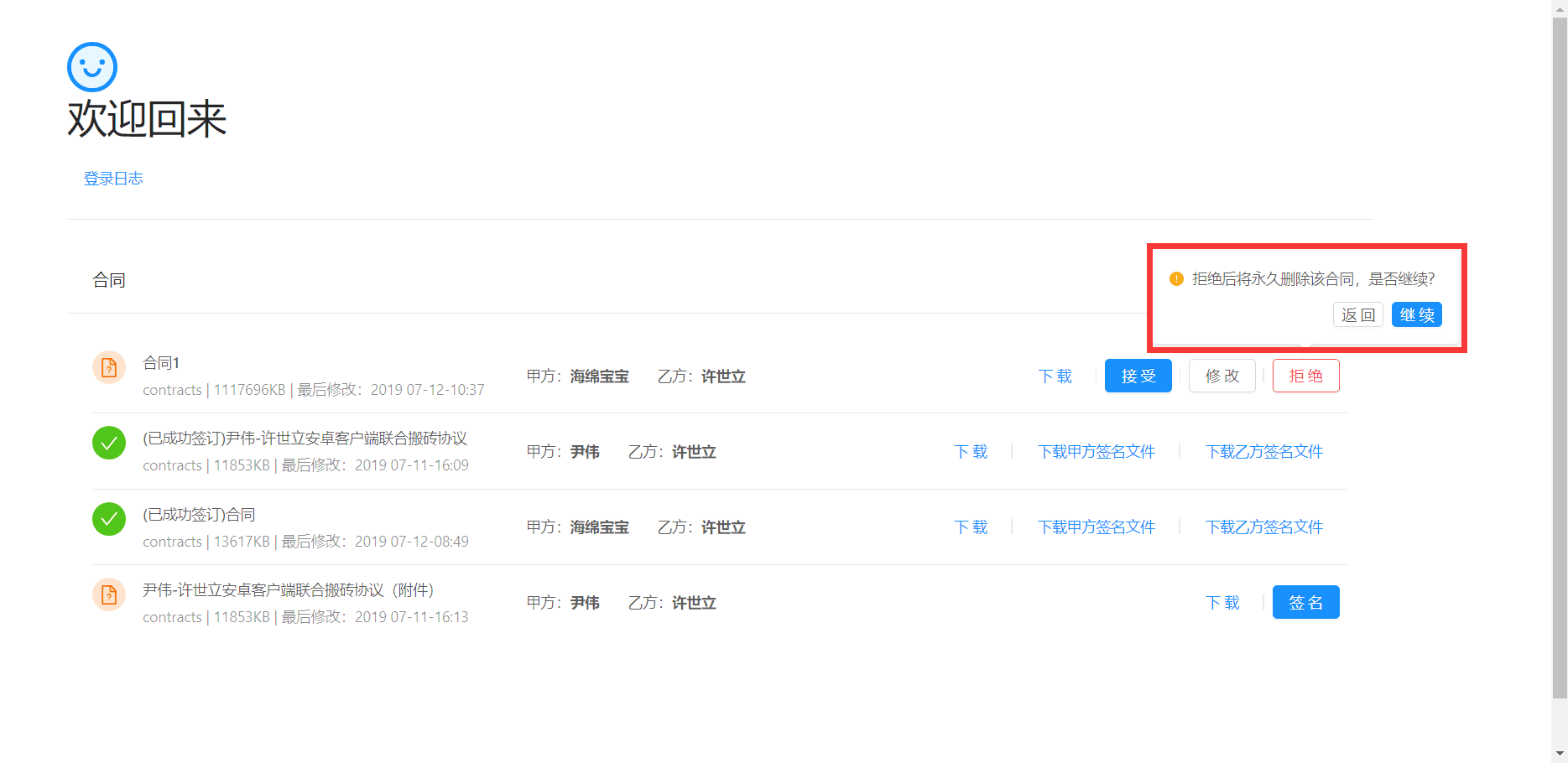
甲方创建成功后等待乙方确认，点击<下载>可查看合同内容，<修改>可重新上传合同进行修改



5、合同确认

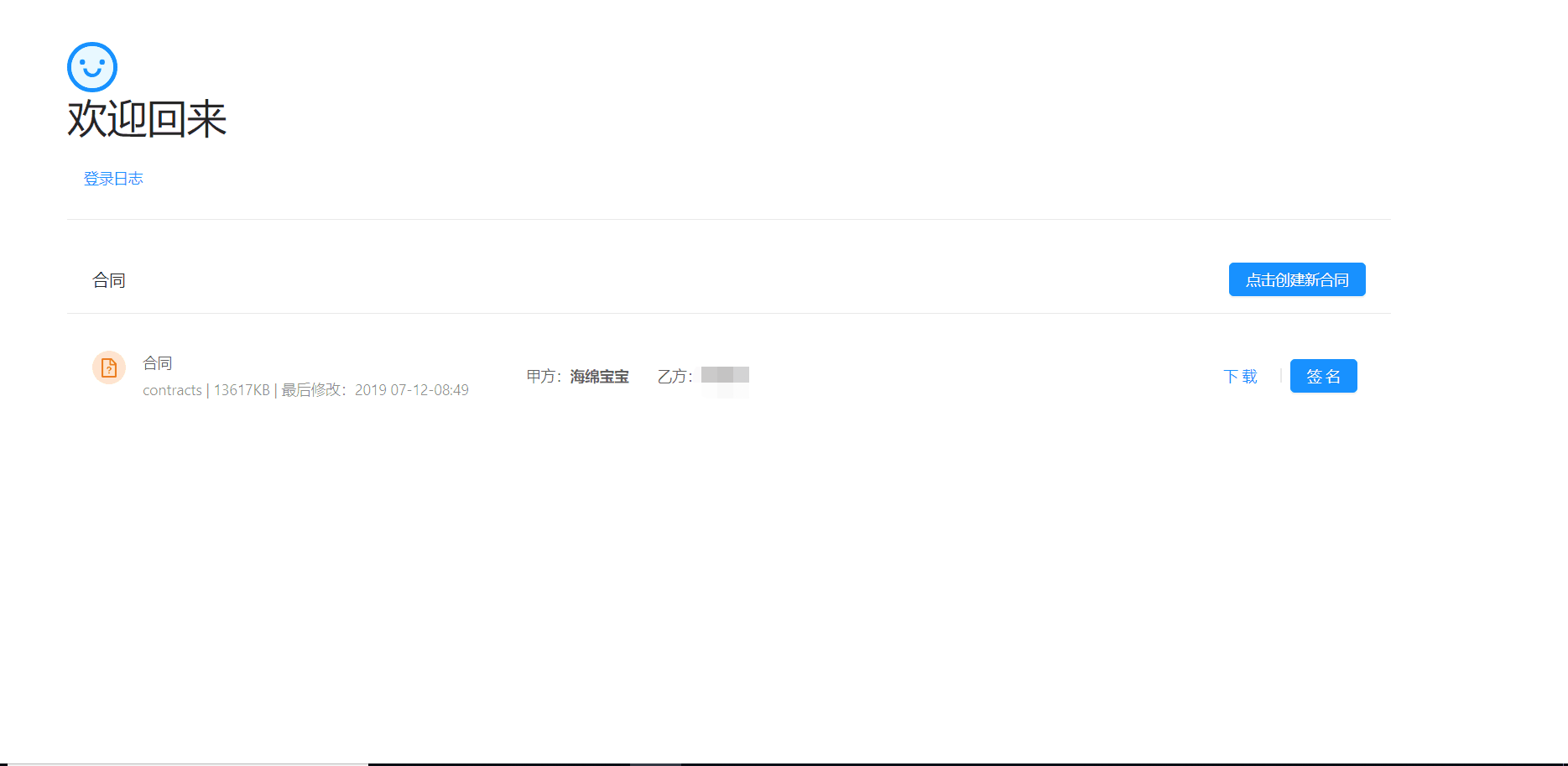
乙方扫码登录之后进入合同页面，可对甲方上传的合同进行<下载>查看，若有需要修改的地方点击<修改>再次上传，可选择<接受>或<拒绝>合同





6、合同签名

乙方确认合同之后，甲方可点击<签名>对合同进行签名，还可点击<下载>查看合同





7、甲方签名

甲方点击<签名>之后登录手机App进行扫码签名



8、等待乙方签名

甲方签名之后等待乙方进行签名



9、双方签名完成

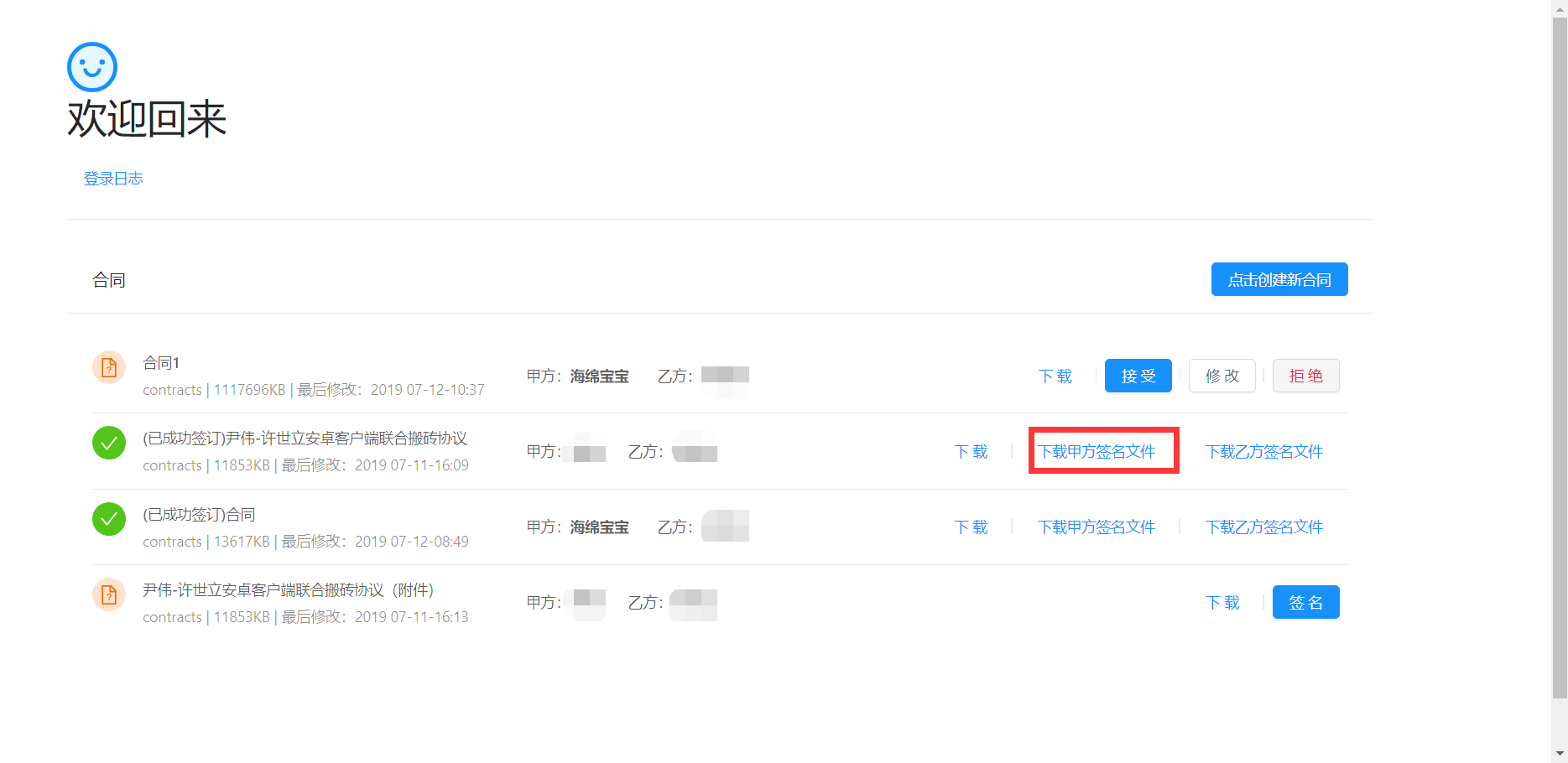
乙方签名完成之后甲方显示：

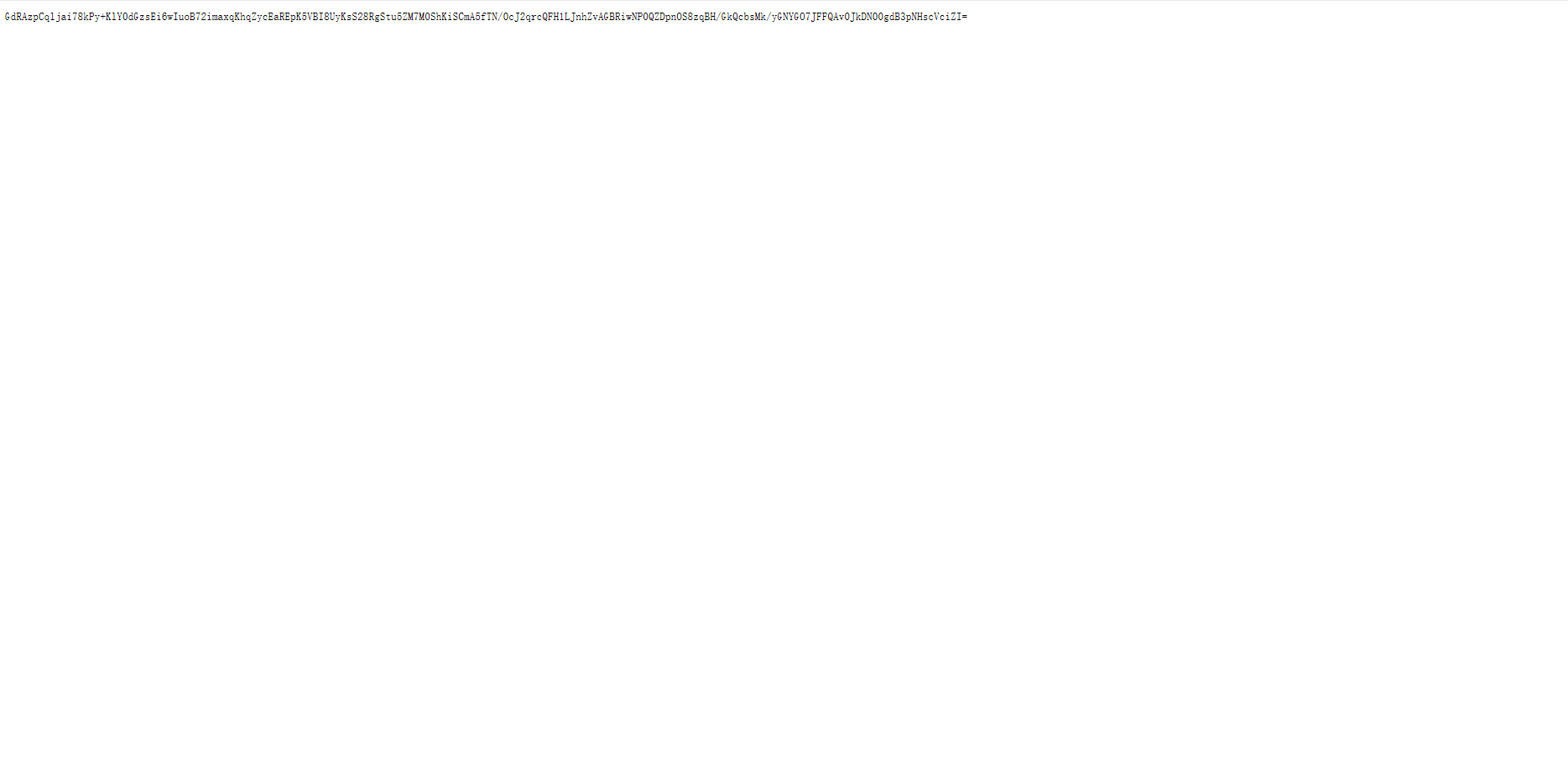


乙方显示：

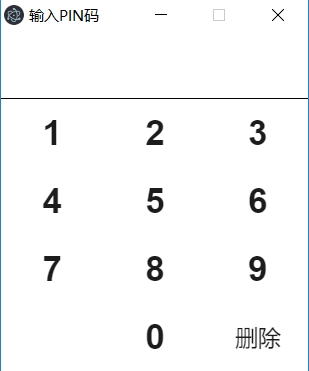


10、下载签名文件





**Web管理员端**



1、登录页

输入管理员PIN码即可登录



2、用户审批

刷新之后对申请注册用户进行审批，可选择<同意>或<拒绝>

# 操作指南

## 快速使用指南

快速描述产品如何操作，如客户需要使用管理工具，首先介绍管理工具各模块主要作用，让客户能够快速理解。

用户注册登录模块

1. 首次使用前需要注册新用户，点击注册按钮进行注册。
2. 填写相关注册信息核验无误之后成功注册，牢记账户名和口令。
3. 再次使用同一设备登录时输入注册时的账号和口令进行登录；若是在新设备上进行登录则选择二维码登录，使用已登录的设备进行扫码登录，登录成功后跳转至用户中心模块。

扫码模块

1. 登录成功后进入这一模块，可使用扫一扫功能。
2. 扫描其他设备二维码进行授权新设备登录。
3. 扫描Web页面显示二维码进行Web授权登录。
4. 扫描Web页面电子合同签名二维码进行签名认证。

电子合同签署模块

用户A和用户B签署合同

1）用户A登录web端，发起一个合同过程。

2）用户A在本地编辑好一个合同(.txt .pdf 等文件)，上传到服务器，点击确认合同。

3）用户B从服务器下载合同，检查或修改合同内容，更新到服务器。点击确认合同。

4）由于用户B修改了合同内容，用户A需要重新确认合同，如果A对合同的内容有异议，可以下载，修改，上传合同并点击确认，同理，如果A修改了合同，B也需要重新确认，直到A和B都在没有修改合同的情况下点击了确认，合同内容最终达成一致，变为不可编辑状态。

5）用户可以在浏览器点击给合同签名（签名过程需要手机扫码），当A和B都对合同签过名之后，成功的生成完整的电子合同。

6用户A和B均可在WEB端一键下载自己签署过的合同。

7）任何用户可以向服务器上传一个完整的电子合同文件来验证合同的有效性。

## 编译与开发环境

开发过程中采用前后端分离的方式。

### 安卓工程的构建

Android文件内保管的是安卓客户端的实现，建议直接以Android Studio打开该工程。

* 编译前，必须设置ANDROID\_HOME环境变量。
* 编译时，应保证采用了项目配置的Gradle脚本。

安卓开发环境：

compileSdkVersion = 28

buildToolVersion = 28.0.3

minSdkVersion = 23

targetSdkVersion = 28

versionCode 1

versionName "1.0"

开发所需依赖：

implementation 'com.google.zxing:core:3.3.0'

implementation 'com.google.android.material:material:1.0.0'

implementation 'com.android.support.constraint:constraint-layout:1.0.2'

implementation 'org.java-websocket:Java-WebSocket:1.4.0'

### Web前端工程的构建

前端分为编辑合同网页和管理员客户端两部分：

* 合同网页

WebFE文件夹内保管的是合同网页的实现。 合同页面使用vue-cli脚手架完成工程的搭建，并使用vuerouter来承载路由的任务。

* 管理员客户端

admin-client文件夹内保管的是管理员客户端的实现。 管理员客户端使用electron提供代码运行的容器，并使用create-react-app构建代码。

开发工具：

Windows10\Jetbrains Webstrom2019.1\nodejs10.13.0\electron\VueJs\AntDesign3.19.6\axios0.19.0\react16\react-router5.0.1\stroybook

运行环境：Windows10

开发所需要依赖

"antd": "^3.19.6",

"axios": "^0.19.0",

"dateformat": "^3.0.3",

"http-proxy-middleware": "^0.19.1",

"node-sass": "^4.12.0",

"prop-types": "^15.7.2",

"qrcode": "^1.3.3",

"react": "^16.8.6",

"react-dom": "^16.8.6",

"react-redux": "^7.1.0",

"react-router-dom": "^5.0.1",

"react-scripts": "3.0.1",

"redux": "^4.0.1"

"@babel/core": "^7.4.5",

"@storybook/addon-actions": "^5.1.9",

"@storybook/addon-links": "^5.1.9",

"@storybook/addons": "^5.1.9",

"@storybook/react": "^5.1.9",

"file-loader": "^4.0.0"

"electron": "^5.0.6",

"electron-builder": "^20.44.4",

"typescript": "^3.5.2"

"classnames": "^2.2.6",

"crypto-js": "^3.1.9-1",

"node-sass": "^4.12.0",

"qs": "^6.7.0",

### 4.2.3 Web后端工程的构建和部署

Server文件夹内保管的是服务端的实现。

* 后端采用SpringBoot框架。
* 将server中的代码copy至新项目，将项目打成jar包。
* 将jar包上传到服务器，然后命令行输入nohup java –jar (jar包名称) &，后台就可以启动该项目。

开发工具：Windows10、IntelliJ IDEA、JDK 11、springBoot2.1.6.RELEASE、  
运行工具：Ubuntu18.04、mysql 5.7.26、tomcat9.0.21、

开发所需要依赖：

springBoot相关依赖：  
spring-boot-starter-thymeleaf 2.1.6.RELEASE  
gson 2.8.5  
spring-boot-starter-jdbc  
spring-boot-starter-web  
jjwt 0.9.0  
spring-boot-starter-websocket 1.3.5.RELEASE  
mysql-connector-java  
spring-boot-devtools  
xmlunit-core  
mybatis-spring-boot-starter  
commons-lang3 3.4  
org.liveSense.fragment.sun.misc 1.0.5  
jstl 1.2  
springBoot插件：  
spring-boot-maven-plugin  
spring-boot-maven-plugin

# 附录A 技术规格

如密码卡则注明电源电压、硬件配置参数等电器特性。

|  |  |  |  |
| --- | --- | --- | --- |
|  | CPU | 内存 | 系统 |
| 服务器 | 1GHz 以上 | 2GB以上 | Windows 10 |
| 客户端 | 1GHz 以上 | 2GB以上 | Android 5.0及以上版本 |