人脸核酸项目后端开发文档

——一阶段

## 简介：

此后端开发用于支持人脸核酸项目的并发服务部分。整个开发分为三个总体阶段，第一阶段解决核心需求问题，第二阶段解决性能优化问题，第三阶段解决部署和安全问题。本次文档描述了第一阶段的开发，目的是明确开发需求统筹开发进度，以便项目组实现基本项目需求。

## 目标受众：

第一阶段开发及测试人员，即项目并发组

## 术语和缩略语：

暂无

## 系统架构

服务器软件：Tomcat（为springboot内置）

软件架构：springboot集成mybatispluss和redispluss。其中mybatispluss用于完成MySQL数据库的管理，redispluss用于提供后端服务以及控制消息队列。

组件关系：Tomcat作为项目底层，控制服务器的运行。Springboot框架负责项目内部各种功能对象的创建和管理，redis作为缓存服务在线对象并处理并发性请求，在需要请求数据时调用mybatis组件从而调用MySQl数据库。

层次关系：

Config：用于配置组件springmvc、mybatispluss、redispluss以及web服务，会重写一些基本配置方法。

Controller：用于向前端提供后端服务的api接口，向下调用service层的服务和dto层中间层数据，同时也会使用一些自定义的工具。

Dao：数据库访问层接口，继承Basemapper提供对entity的基本方法的调用接口。

Dto：数据传输对象，用于在不同层之间提供数据对象，方便各层传输数据。

Entity：数据库表的映射实体类，受dao层的调用，处于控制数据库最底层。

Service:接口组继承IServce之外，提供一些其它业务层方法接口。接口供controller层调用。实现组（impl）实现接口组的方法，并重写Iservce内部提供的相关方法接口。

Uils：工具类，用于生成工具对象以供各个层次的对象调用，包括redis的数据处理，数据序列化，数据格式化，数据安全操作等。

Application：主程序，启动所有服务。

Resources：用于存放各种资源，如数据库构建源文件，yaml配置，lua脚本（用于数据初始化、配置管理、任务调度、集群）等。

## 功能用例：

1. 普通用户业务

**用户注册**：

一般：点击注册按钮——填写姓名——填写身份证号——填写密码——确认密码——点击确认——注册完成——进行用户登录跳转

无效身份证号码：点击注册按钮——填写姓名——填写身份证号——填写密码——确认密码——点击确认——提示身份证号码无效

密码格式错误：点击注册按钮——填写姓名——填写身份证号——填写密码——提示密码格式错误——等待用户重新输入密码

重复注册：点击注册按钮——填写姓名——填写身份证号——填写密码——确认密码——点击确认——提示用户已存在

（请求失败）：点击注册按钮——填写姓名——填写身份证号——填写密码——确认密码——点击确认——提示请求失败并返回失败信息

取消：点击注册按钮——在确认前任何一步可以取消

**用户登录**：

一般：点击登录按钮——填写身份证号码——填写密码——点击确认——登录成功页面跳转

密码错误：点击登录按钮——填写身份证号码——填写密码——点击确认——提示密码错误——等待重新输入密码

用户不存在：点击登录按钮——填写身份证号码——填写密码——点击确认——提示用户不存在——等待重新输入用户和密码

（请求失败）：点击登录按钮——填写身份证号码——填写密码——点击确认——提示请求失败并返回失败信息

取消：点击登录按钮——在确认前任何一步可以取消

**人脸录入**：

一般：点击人脸录入按钮——进行拍照——点击确认——人脸录入完成

（请求失败）：点击人脸录入按钮——进行拍照——点击确认——提示请求失败并返回失败信息（登录状态失效等）

取消：点击人脸录入按钮——在确认前任何一步可以取消

**人脸修改**：

一般：点击人脸信息记录——选择修改——输入密码——进行拍照——点击确认——人脸修改完成

密码错误：点击人脸信息记录——选择修改——输入密码——提示密码错误并等待重新输入

（请求失败）：点击人脸信息记录——选择修改——输入密码——进行拍照——点击确认——提示请求失败并返回失败信息（登录状态失效等）

**人脸删除**：

一般：点击人脸信息记录——选择修改——输入密码——点击确认——人脸删除完成

密码错误：点击人脸信息记录——选择修改——输入密码——提示密码错误并等待重新输入

（请求失败）：点击人脸信息记录——选择修改——输入密码——点击确认——提示请求失败并返回失败信息（登录状态失效等）

取消：点击人脸信息记录——选择修改——在点击确认前可以取消

**信息修改**：

一般：点击信息修改按钮——输入密码——选择修改的信息——更改信息——点击确认——信息修改完成

密码错误：点击信息修改按钮——输入密码——提示密码错误并等待重新输入

（请求失败）：点击信息修改按钮——输入密码——选择修改的信息——更改信息——点击确认——提示请求失败并返回失败信息（登录状态失效等）

取消：点击信息修改按钮——在点击确认前可以取消

**核酸结果查询**：

一般：点击核酸结果查询——返回14天内核酸结果

（请求失败）：点击核酸结果查询——提示请求失败并返回失败信息（登录状态失效等）

1. 核酸检测业务

**登记终端注册**：

一般：点击登记终端注册——生成账号——输入密码——二次确认密码——点击确认——注册完成

密码格式非法：点击登记终端注册——生成账号——输入密码——提示密码格式非法并等待重新输入

（请求失败）：点击登记终端注册——生成账号——输入密码——二次确认密码——点击确认——提示请求失败并返回失败信息

取消：点击登记终端注册——生成账号——在确认前可取消

**登记终端登录**：

一般：点击登记终端登录——输入账号——输入密码——点击确认——登录成功——页面跳转

终端账号不存在：点击登记终端登录——输入账号——输入密码——点击确认——提示账号不存在——等待重新输入

密码错误：点击登记终端登录——输入账号——输入密码——点击确认——提示密码错误——等待重新输入

（请求失败）：点击登记终端登录——输入账号——输入密码——点击确认——提示请求失败并返回失败信息

取消：点击登记终端登录——在确认前可取消

**扫码机登录**：

一般：扫码机开机——连接网络——发送登录请求——返回登录成功

网络中断：扫码机开机——连接网络——连接失败——重新尝试并提示网络终端

请求超时：扫码机开机——连接网络——发送登录请求——请求超时——再次发送

重复请求：扫码机开机——连接网络——发送登录请求——返回请求重复

**扫码机与终端配对**：

一般：登录终端中点击扫码机配对——输入扫码机序列号——点击配对——配对成功

序列号空：登录终端中点击扫码机配对——输入扫码机序列号——点击配对——提示序列号为空并等待重新输入

已有配对：登录终端中点击扫码机配对——提示已有配对是否修改——可选修改与取消

（请求失败）：登录终端中点击扫码机配对——输入扫码机序列号——点击配对——提示请求失败并返回失败信息

取消：登录终端中点击扫码机配对——在确认前可取消

**查看配对扫码机状态**：

一般：登录终端中点击查看状态——返回扫码机登录状态、当前管号、剩余可用

没有配对：登录终端中点击查看状态——提示没有配对

（请求失败）：登录终端中点击查看状态——提示请求失败并返回失败信息

**获取管号**：

一般：接收到扫描命令——扫描——返回信息（管号、序列号、剩余可用）到服务器——收到确认信息

连接弱：接收到扫描命令——扫描——返回信息（管号、序列号、剩余可用）到服务器——未收到确认信息或超时——再次发送并注明二次

无管：接收到扫描命令——扫描——发现空管——返回空管信息并声光提示——收到确认信息

扫描错误：接收到扫描命令——扫描——发现错误（管号非法、活动件故障等）——返回错误信息并声光提示——收到确认信息

**创建结果表项**：

一般：受到来自扫码器的管号信息——检查管号——添加管号到结果表——向扫码机返回确认信息——向登记终端返回新管号和可用数量

非法管号：受到来自扫码器的管号信息——检查管号——发现管号非法——向扫码机提出扫描命令——向登记终端返回错误信息

操作失败：受到来自扫码器的管号信息——检查管号——添加管号到结果表——操作失败——向登记终端返回错误信息——在运维终端输出错误提示

**人脸图像获取**：

暂略

**人脸特征向量获取**：

暂略

**身份信息获取**：

暂略

**核酸订单生成**：

一般：完成扫脸——点击身份确认——获取身份信息——获取管号——获取时间——获取生成的核酸订单号——发送请求信息——收到确认消息

身份错误：完成扫脸——身份错误——选择重新扫脸或者输入身份证号——返回一般

身份识别请求失败：完成扫脸——身份识别失败——选择重新扫脸或者输入身份证号——返回一般

管号不可用：完成扫脸——点击身份确认——获取身份信息——获取管号——发现管号不可用——发送获取管号请求——提示等待——受到管号——返回一般的获取时间

请求失败或者超时：完成扫脸——点击身份确认——获取身份信息——获取管号——获取时间——获取生成的核酸订单号——发送请求信息——请求失败或超时——提示错误信息

重复请求：完成扫脸——点击身份确认——获取身份信息——获取管号——获取时间——获取生成的核酸订单号——发送请求信息——收到重复请求信息——回滚管号状态——提示重复登记

1. 医疗机构业务（此业务为演示内容，设计并不不完善）

**医疗机构注册**：

一般：点击注册——输入医疗机构名称——生成账号——输入密码——二次确认密码——点击确认——注册成功

取消：点击注册——在点击确认前可取消

（请求失败）：点击注册——输入医疗机构名称——生成账号——输入密码——二次确认密码——点击确认——提示请求失败并返回错误信息

**医疗机构登录**：

一般：点击登录——输入账号——输入密码——点击确认——登录成功

取消：点击登录——在点击确认前可取消

账号非法：点击登录——输入账号——输入密码——点击确认——提示账号非法并等待重新输入

密码错误：点击登录——输入账号——输入密码——点击确认——提示密码错误并等待重新输入

（请求失败）：点击登录——输入账号——输入密码——点击确认——提示请求失败并返回错误信息

**核酸结果表按管号录入结果**：

一般：点击录入结果——输入密码——填写管号、结果表单——点击确认——提示录入成功

密码错误：点击录入结果——输入密码——提示密码错误并等待重新输入

非法管号：点击录入结果——输入密码——填写管号、结果表单——点击确认——提示非法管号并等待非法的管号重新录入

取消：点击录入结果——在点击确认之前可取消

（请求失败）：点击录入结果——输入密码——填写管号、结果表单——点击确认——提示请求失败并返回错误信息

**核酸结果表按管号查询**：

一般：点击查询——输入密码——填写查询管号表单——点击确认——返回结果（包括有、无、非法管号、请求失败信息）

密码错误：点击查询——输入密码——提示密码错误并等待重新输入

## 功能实现

普通用户业务：

登录拦截（UserLoginInterceptor）——在mvcconfig中配置并注入，可设置需要拦截的请求

Prehandle： 拦截请求request，从中取出token，内含登录时设置的token并检查是否为空。以token为key从redis的已登录用户列表中查找已经登录的用户，并以map的格式返回，这个map包含登录时向redis内存储的用户DTO数据。判断map不为空，表明用户存在且已经登录。此时，将map转换成userDTO并加入名为UserHolder的用户线程池中，这个线城池将处理此次用户的请求。并刷新redis内以token为key的记录的有效期。在有效期内，用户被视为已经登录。返回true放行。

afterComletion：拦截请求request，在请求结束后，将处理此次用户请求时创建的用户线程从UserHolder线程池中移除，并执行默认（super）的结束拦截任务。

注册（sign-in）——在usercontroller中以API的形式供前端调用

传入以signForm格式的请求体，取出其中以字符串为格式的各个数据，包括姓名，身份证，电话，密码，密码确认。对身份证，电话，密码进行格式校验，对密码和密码确认进行相等确认后，查看由myatispluss管理的users表中是否有id相同的用户，若有，则提示已经注册，若没有，则新建用户对象，设置各个必要的参数，并存储到数据库内的表中。

登录（login）——在usercontroller中以API的形式供前端调用

传入以loginForm为格式的请求体，取出其中的id和phone，验证格式合法性。并按照电话优先的用户查找数据库，若不存在返回错误信息。若成功获取到了正确的用户对象，则利用用户对象核对登录时输入的密码，不正确返回错误，正确则 生成随机的token，将用户对象转化为没有敏感信息的userDTO的对象， 将DTO对象转换为map，并以token为key存储到redis中，设置这个记录的有效期（有效期的配置在redisconstants中）。并向前端返回包含token的成功信息。

查询14天记录（get-14d-orders）——在registercontroller中以api的形式供前端调用，

在经过登录拦截器的校验过后，传入以getOrderDTO为格式的请求体，取出其中的id，在缓存中查找以id为key的核酸订单缓存记录。若找不到，那么返回错误。找到了，刷新缓存的持续时间，返回记录组（格式？）。

刷新（refresh）——在registercontroller中以api的形式供前端调用

在经过登录拦截器的校验过后，传入以getOrderDTO为格式的请求体，取出其中的id，并以id为key调用mysql数据库，查询符合要求的（14天或者其它要求）记录，返回，并更新缓存，同时刷新缓存持续时间。

新建订单（neworder）——在registercontroller中以api的形式供前端调用，

在经过核酸登记设备登录校验过后，传入以register为格式的请求体，取出其中的id，开始事务，在以id为key的缓存中增加新的这条记录，并刷新缓存持续时间，（可能会调整缓存内的信息），在mysql中添加这条记录，结束事务。返回事务执行的结果。

## 非功能需求：

## 数据库设计：

## 接口设计：

## 测试计划：

## 部署计划：