# 接口设计说明书

# 第一章 引言

## 1.1 设计目标

本请假系统在设计时做到了前后端分离的要求。接口是唯一能直接通过调用Service层的函数实现系统数据的获取、新增、删除与修改的途径。控制器中除了接口函数以为的返回视图的函数不直接调用Service层，而是通过调用接口函数实现相关的功能。

本系统的接口在外部使用get或post请求发送，返回的是json数据返回的json中包含一个state状态码和data数据。

## 1.2 适用范围

本接口设计说明书适用于海宁市人民医院请假系统。本文档的编写目的为规范开发人员的接口设计风格，同时为后续可能会进行的进一步开发安卓端的相应系统提供接口使用指导。

## 1.3 相关术语

API：在本文档中指文档中列出的通过get或者post请求发送参数，从而以json形式返回应答数据的交互方式。

# 第二章 分析设计

### 2.1 接口模块划分

### 2.1.1 用户管理模块

### 2.1.2 请假查看模块

### 2.1.3 流程推进模块

### 2.1.4 部门管理模块

## 2.2 部门管理模块

### 2.2.1 用户管理模块接口

（1）管理员查看用户列表api

@RequestMapping(value = **"/api/list"**, method = RequestMethod.***GET***)  
@ResponseBody  
*//返回所有的用户列表，需要cookie存储的登录信息是具有用户管理权限的用户***public** ResultWrapper<List<User>> listAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify)

（2）管理员查看某用户信息api

@RequestMapping(  
 value = **"/api/{userId}/detail"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//传入用户的id，返回用户信息，需要cookie中有具有管理员权限的用户***public** ResultWrapper<User> getByIdAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId)

1. 登录api

@RequestMapping(  
 value = **"/api/login"**, *//测试用，以后会改成 /login* method = RequestMethod.***POST***, *//测试用，以后会改成POST* produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//传入用户名和密码，如果用户名和密码正确，则设置相应的cookie。返回登录结果状态码***public** ResultWrapper<User> loginAPI(@ModelAttribute(**"username"**) String username,  
 @ModelAttribute(**"password"**) String password,  
 HttpServletResponse response)

1. 判断是否登录api

@RequestMapping(  
 value = **"/api/islogin"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//根据cookie判断是否已登录，返回登录状态码***public** ResultWrapper<User> isLoginAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify)

1. 退出登录api

@RequestMapping(  
 value = **"/api/logout"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//消除登录cookie***public** ResultWrapper<User> logoutAPI(HttpServletResponse response)

1. 编辑用户api

@RequestMapping(  
 value = **"/api/edit"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//编辑用户，更新用户信息，更新的用户的id为传入的id***public** ResultWrapper<User> editAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId,  
 @ModelAttribute(**"userName"**) String userName,  
 @ModelAttribute(**"userSex"**) String userSex,  
 @ModelAttribute(**"userAge"**) **int** userAge,  
 @ModelAttribute(**"userDepartment"**) **int** userDepartment,  
 @ModelAttribute(**"userLeader"**) **int** userLeader,  
 @ModelAttribute(**"userTimeLeft"**) **int** userTimeLeft,  
 @ModelAttribute(**"isAdmin"**) **int** isAdmin)

1. 重置密码api

@RequestMapping(  
 value = **"/api/reset"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//传入id,将该id的用户的密码重置为六个零，需要cookie中有权限为管理员的用户***public** ResultWrapper<User> resetPasswordAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId)

1. 判断当前登录的是否是管理员api

@RequestMapping(  
 value = **"/api/isAdmin"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//判断当前cookie中存储的用户是否具有管理员权限。（三个顶层部门的领导具有管理员权限）***public** ResultWrapper<User> isAdminAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify)

1. 获取所有部门信息api

@RequestMapping(  
 value = **"/api/getAllDepartments"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//获取所有的部门信息，需要当前cookie中是合法的未过期的登录用户***public** ResultWrapper<List<Department>> getAllDepartmentsAPI(@CookieValue(**"userVerify"**) String userVerify)

1. 删除用户api

@RequestMapping(  
 value = **"/api/delete"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//输入用户id，删除该id的用户，需要cookie中存储的是具有管理员权限的用户***public** ResultWrapper<User> deleteAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId)

1. 新增用户api

@RequestMapping(  
 value = **"/api/add"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//添加一个新用户，需要cookie中是具有管理员权限的用户***public** ResultWrapper<User> addAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userUsername"**) String userUsername,  
 @ModelAttribute(**"userName"**) String userName,  
 @ModelAttribute(**"userSex"**) String userSex,  
 @ModelAttribute(**"userAge"**) **int** userAge,  
 @ModelAttribute(**"userDepartment"**) **int** userDepartment,  
 @ModelAttribute(**"userLeader"**) **int** userLeader,  
 @ModelAttribute(**"userTimeLeft"**) **int** userTimeLeft,  
 @ModelAttribute(**"isAdmin"**) **int** isAdmin)

1. 判断输入的id是否是登录者api

@RequestMapping(  
 value = **"/api/isMyself"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//输入一个id，输出这个id与当前cookie中的用户是否是同一个人***public** ResultWrapper<User> isMyselfAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId)

1. 编辑本人信息api

@RequestMapping(  
 value = **"/api/editMyself"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//编辑自己的信息，需要cookie中具有已登录的用户***public** ResultWrapper<User> editMyselfAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userName"**) String userName,  
 @ModelAttribute(**"userSex"**) String userSex,  
 @ModelAttribute(**"userAge"**) **int** userAge)

1. 修改本人密码api

@RequestMapping(  
 value = **"/api/changePassword"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//输入用户id、旧密码和新密码,更改密码，需要输入的id与cookie中存储的登录用户是同一个人***public** ResultWrapper<User> changePasswordAPI(@CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"userId"**) **int** userId,  
 @ModelAttribute(**"oldPassword"**) String oldPassword,  
 @ModelAttribute(**"newPassword"**) String newPassword)

### 2.2.2 请假查看模块

1. 查看自己的所有请假信息api

@RequestMapping(  
 value = **"/api/myApplicationList"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//查看自己的所有请假申请，需要cookie中存储有登录的用户***public** ResultWrapper<List<Form>> myApplicationList(  
 @CookieValue(value = **"userVerify"**, required = **false**) String userVerify)

1. 查看所有已经通过的请假申请（预留给财务部门的接口）

@RequestMapping(  
 value = **"/api/showAllPassedApplications"**,  
 method = RequestMethod.***GET***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//查看所有已经通过的请假申请，需要cookie中是有管理员权限的用户***public** ResultWrapper<List<Form>> showAllPassedApplications(  
 @CookieValue(value = **"userVerify"**, required = **false**) String userVerify)

### 2.2.3 流程推进模块

### 2.2.4 部门管理模块

1. 添加新部门api

@RequestMapping(  
 value = **"/api/addDepartment"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//添加新的部门，需要cookie中存储有管理员权限的用户***public** ResultWrapper<Department> addDepartment(  
 @CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"departmentName"**) String departmentName,  
 @ModelAttribute(**"departmentParent"**) **int** departmentParent)

1. 删除部门api

@RequestMapping(  
 value = **"/api/deleteDepartment"**,  
 method = RequestMethod.***POST***,  
 produces = {**"application/json;charset=UTF-8"**}  
)  
@ResponseBody  
*//传入用户id，删除该id的部门，要求该部门内无员工且是三个顶层部门以外的部门，需要cookie中存储有具有管理员权限的用户***public** ResultWrapper<Department> deleteDepartment(  
 @CookieValue(value = **"userVerify"**, required = **false**) String userVerify,  
 @ModelAttribute(**"departmentId"**) **int** departmentId)