

心率分析系统需求

及设计文档

**小组成员：**

16126167 李鹏翔

# 概述

## 1.1 编写目的

本文档是基于与客户进行多次QA问答及确认编写的。本文档的编写为下阶段的设计、开发提供依据，为项目组成员对需求的详尽理解，以及在开发过程中的协同工作提供强有力的保证。同时，本文档也作为项目评审验收的依据之一。

## 1.2 系统目标

本系统有两类主要目标人群：**普通用户以及医生。**

本系统为**普通用户**提供实时查看自己健康状况的功能。病人在该系统中注册成为普通用户，登陆系统后可以及时传送自己的心率数据，通过系统分析后查看自己的健康状况，还可以查看历史健康状况。

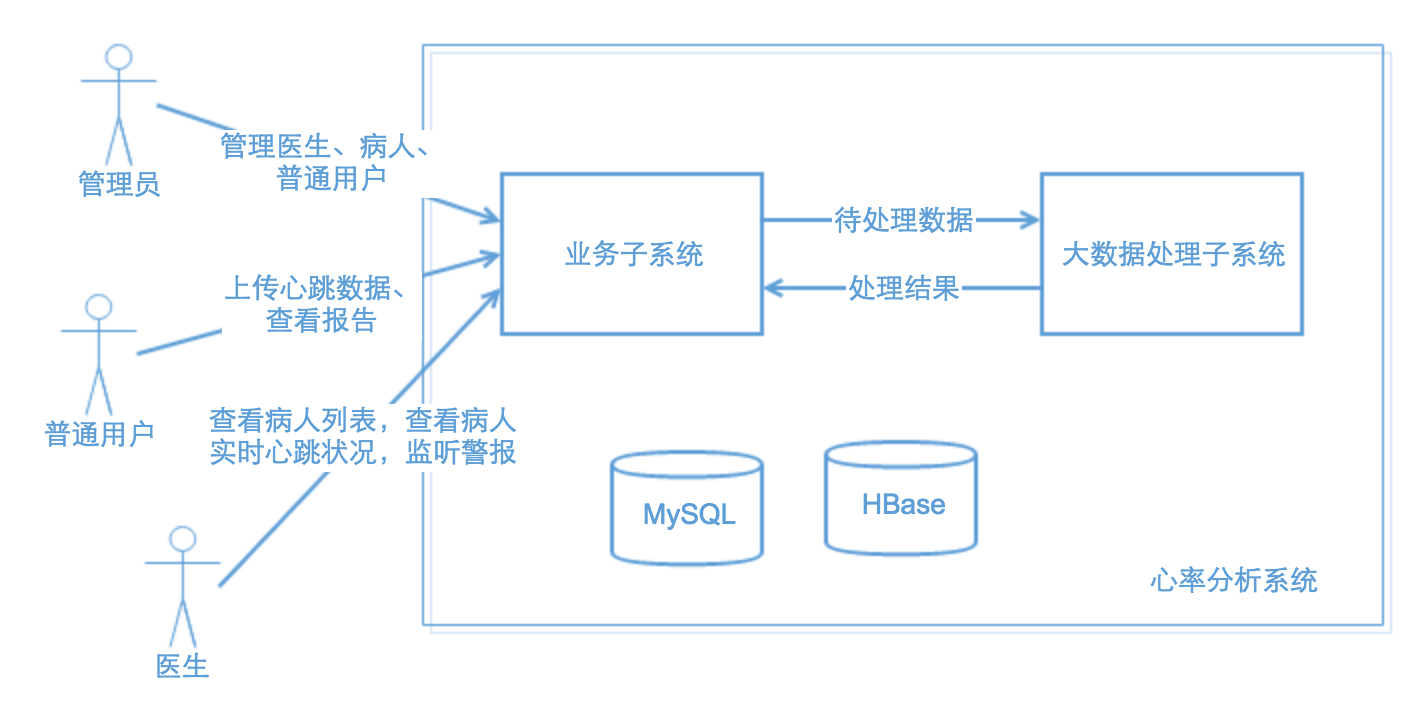
本系统为**医生**提供一个实时监控病人健康状况的功能。医生登陆此系统后可以查看病人列表，进而监控每个病人的心率数据和查看病人的健康状况。

# 利益相关者分析

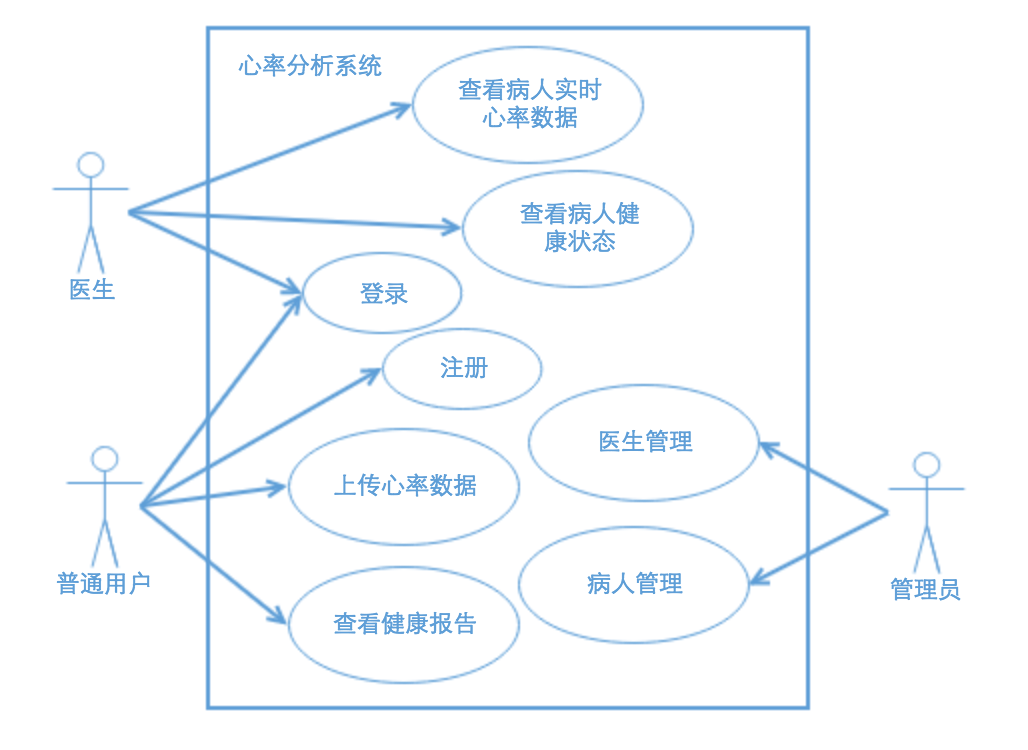
该系统的利益相关者有3个，分别是普通用户、医生和系统管理员。从他们的角度来分析，我们确定了关键要求。如下：



# 系统边界



# 系统用例



用例说明：

1. **医生**指的是在医院工作的治疗病人的医务人员，他们需要通过系统查看他的病人的实时心率数据，而且需要查看该病人的健康状况，医生的进行上述操作的前提是他必须能够登录系统，并且能够查看病人列表；
2. **普通用户**指的是一般用户，他们想借此系统检测自己的健康状况。首先他们需要注册系统，再登录系统，然后他们可以上传自己的心率数据，通过系统的分析，他可以查看自己的报告。除此之外，他也可以查看自己的历史健康报告；
3. **管理员**指的是此系统具有最高权限的系统维护人员，主要任务是辅助系统的运行。首先，由于医生没有注册权限，所有管理员需要为医生新建账号，然后对医生信息进行管理，然后病人入院治疗时需要录入病人信息，以及后续对病人信息的管理。

# 功能性需求

功能性需求主要描述的是系统的功能，面向的是最终用户。下面以不同的用户划分不同的子系统，分析系统的功能，为系统设计部分提供参考。

## 5.1 普通用户子系统

### 5.1.1 用例

用户登录系统后，可以上传自己的心跳数据，上传成功后，后台的数据分析模块进行分析，将分析结果存入数据库。用户点击“查看报告”可以查询所有上传数据的分析结果，按时间先后显示。



Figure 5.1 普通用户用例图

### 5.1.2 系统用例陈述

Table 5.1 用例描述

|  |  |
| --- | --- |
| 用例一 |  |
| 用例名 | 上传心跳数据 |
| 参与者 | 病人、系统 |
| 描述 | 该用例为用户上传个人心跳数据，检测身体健康状况 |
| 前置条件 | 用户已成功登录系统 |
| 后置条件 | 无 |
| 触发条件 | 点击“上传”按钮 |
| 基本流程 | 1、点击“上传”按钮  2、选择数据文件 |
| 结束 | 出现“上传成功，数据分析中”提示框 |

Table 5.2 用例描述

|  |  |
| --- | --- |
| 用例二 |  |
| 用例名 | 查看健康报告 |
| 参与者 | 病人、系统 |
| 描述 | 该用例为用户上传完数据后，查看后台健康分析结果 |
| 前置条件 | 用户已经上传心跳数据 |
| 后置条件 | 无 |
| 触发条件 | 用户点击“查看报告”按钮 |
| 基本流程 | 1、点击“查看”按钮  2、选择查看某次心跳数据 |
| 结束 | 显示健康分析结果 |

### 5.1.3 系统流程图



Figure 5.2 普通用户流程图

## 5.2 医生子系统

### 5.2.1 医生子系统用例

如图5.3所示，医生共有4个用例：全部病人健康监控，查看单个病人实时心跳数据及健康状态，非健康状态报警以及医生诊断病人健康状态。首先，医生可以查看该医生的全部病人的实时状态，显示健康或者不健康，同时医生可以选择某位病人进行重点监控，此时显示的是病人的实时心跳数据以及健康状态，同样分为健康状态和非健康状态。无论以上哪种方式对病人进行监控，一旦病人出现非健康状态，系统会自动报警，医生可针对报警病人进行诊断，并将诊断结果在系统上进行填报。



Figure 5.3 医生子系统用例图

### 系统用例陈述

以下表格是针对5.2.1的医生子系统用例图进行详细的用例描述，主要包含用例名、参与者、用例描述、用例的前置和后置条件、用例的触发条件、基本流程以及用例的结束标志。

Table 5.3 用例描述

|  |  |
| --- | --- |
|  |  |
| 用例名 | 非健康状态报警 |
| 参与者 | 医生，病人 |
| 描述 | 该用例主要是病人心跳处于非健康状态时报警通知医生 |
| 前置条件 | 病人的实时心跳数据输入 |
| 后置条件 | 如果心跳数据不健康向医生发出报警信号 |
| 触发条件 | 病人心跳数据经过分析后处于非健康状态 |
| 基本流程 | 1、实时收集病人心跳数据  2、分析病人心跳数据  3、如若病人心跳非健康发出报警  4、医生接到报警信号 |
| 结束 | 医生确认报警信号后此用例结束 |

Table 5.4 用例描述

|  |  |
| --- | --- |
|  |  |
| 用例名 | 全部病人健康监控 |
| 参与者 | 医生，病人 |
| 描述 | 该用例主要是展示所有病人当前的健康状况 |
| 前置条件 | 病人的实时心跳数据输入 |
| 后置条件 | 通过状态按钮显示病人健康状况 |
| 触发条件 | 医生浏览全部病人信息页面时 |
| 基本流程 | 1、实时收集病人心跳数据  2、分析病人心跳数据  3、区分病人的健康程度并通过状态按钮进行展示  4、医生查看状态按钮 |
| 结束 | 医生离开此页面 |

Table 5.5 用例描述

|  |  |
| --- | --- |
|  |  |
| 用例名 | 查看单个病人实时心跳数据以及健康状态 |
| 参与者 | 医生，病人 |
| 描述 | 该用例主要是展示某位病人当前的健康状况 |
| 前置条件 | 病人的实时心跳数据 |
| 后置条件 | 病人的心电图和健康分析信息 |
| 触发条件 | 医生浏览单个病人信息 |
| 基本流程 | 1、实时收集病人心跳数据  2、分析病人心跳数据  3、以实时更新的心电图来展示数据，并展示分析结果  4、医生在界面中看到以上信息 |
| 结束 | 医生离开此页面 |

Table 5.6 用例描述

|  |  |
| --- | --- |
|  |  |
| 用例名 | 医生诊断病人健康状态 |
| 参与者 | 医生，病人 |
| 描述 | 该用例主要是医生根据报警信息进行诊断并填写诊断结果 |
| 前置条件 | 病人的实时心跳数据 |
| 后置条件 | 医生的诊断 |
| 触发条件 | 医生收到报警 |
| 基本流程 | 1、医生收到报警信息  2、医生根据这段时间病人的心电数据写出自己的诊断  3、新增记录自动添加到数据集 |
| 结束 | 医生上传完数据 |

### 5.2.3 系统流程图

如图5.4是医生子系统的流程图，医生进入系统之后，首先展示全部病人的心跳健康状态，并且可以通过点击具体病人查看该病人具体的心跳数据及健康状态，一旦系统对某位病人发出报警信息，医生需对该病人诊断，并在诊断结束之后，到系统对病人实际状态进行填写。



Figure 5.4 医生子系统流程图

## 5.3 后台（管理员）子系统

### 5.1.1 用例

用户登录系统后，可以添加或者删除医生账户，也可以给医生添加其负责的病人。



Figure 5.5 管理员子系统用例图

### 5.1.2 系统用例陈述

Table 5.7 用例描述

|  |  |
| --- | --- |
| 用例一 |  |
| 用例名 | 管理医生账户 |
| 参与者 | 病人、医生和系统 |
| 描述 | 该用例为后台管理员添加医生账号，医生只有被添加后才能登陆系统 |
| 前置条件 | 管理员已成功登录系统 |
| 后置条件 | 无 |
| 触发条件 | 点击“上传”按钮 |
| 基本流程 | 1、输入医生工号  2、点击“添加” |
| 结束 | 新添加的医生出现在医生列表中 |

Table 5.8 用例描述

|  |  |
| --- | --- |
| 用例二 |  |
| 用例名 | 管理医生与病人的关系 |
| 参与者 | 病人、医生和系统 |
| 描述 | 该用例为后台管理员为已在系统中的医生添加其负责的病人，维护病人与医生的关系 |
| 前置条件 | 医生已在列表中 |
| 后置条件 | 无 |
| 触发条件 | 点击“添加病人”按钮 |
| 基本流程 | 1、输入病人号  2、选择对应的医生  3、点击“完成” |
| 结束 | 该病人在医生负责的病人列表中 |

### 5.1.3 系统流程图



Figure 5.6 管理员系统流程图

# 非功能性需求

非功能性需求衡量是一款产品好坏的一项重要指标，下面就一下方面简单描述。

可用性：系统需要保证前面的功能性需求全部可用并且能够正确执行。

易用性：系统需要有个良好的用户接口，让用户易于操作。

安全性：系统需要采用身份认证的方式，保证数据安全性。

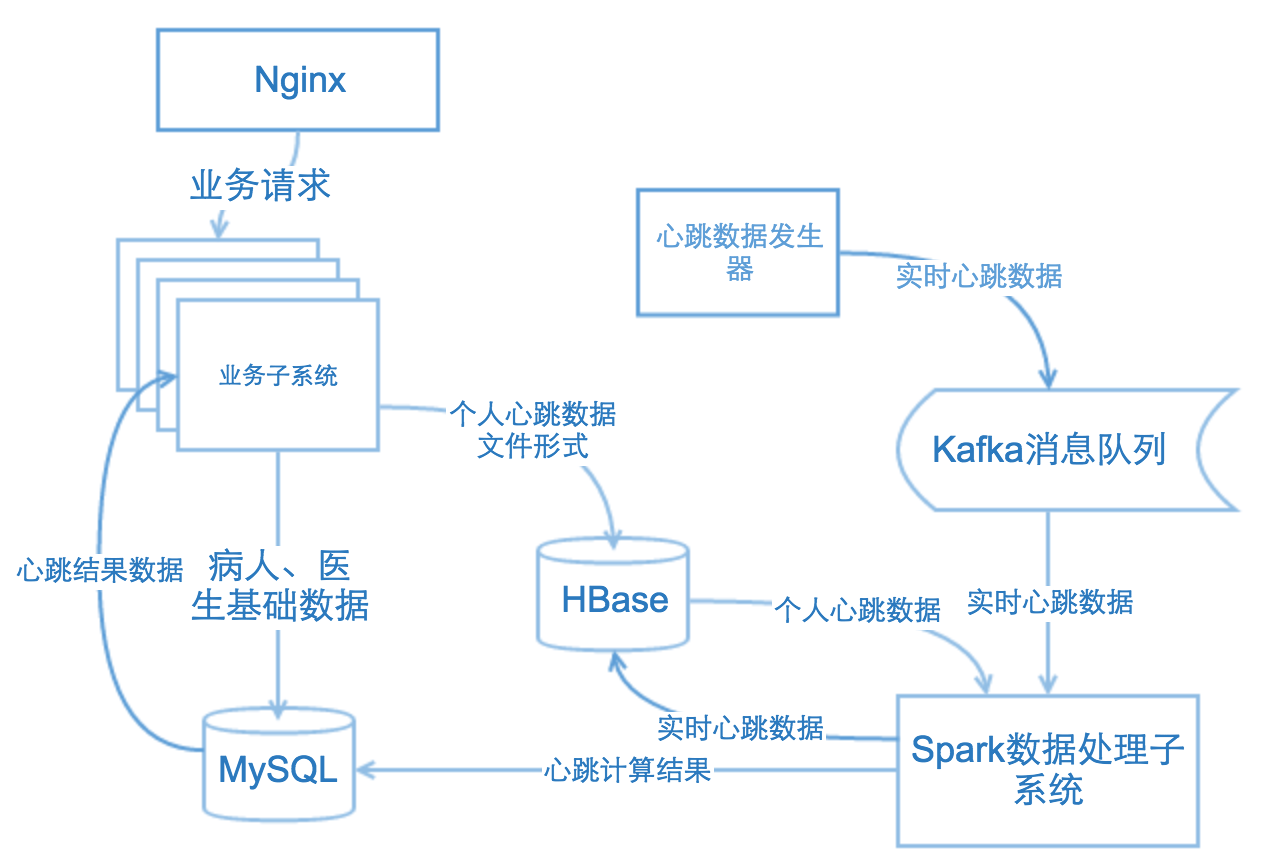
可扩展性：系统在设计的同时需要考虑将来的扩展，以便于使系统更加灵活。

可维护性：系统需要遵循一定的开发规范，以便于让后续的维护工作更加简单。

# 系统架构

系统架构部分主要从设计的角度描述系统，这是系统开发的主要参考，通过系统架构的设计，能对系统有一个整体的把握，下面将从部署架构和逻辑架构两方面来描述。

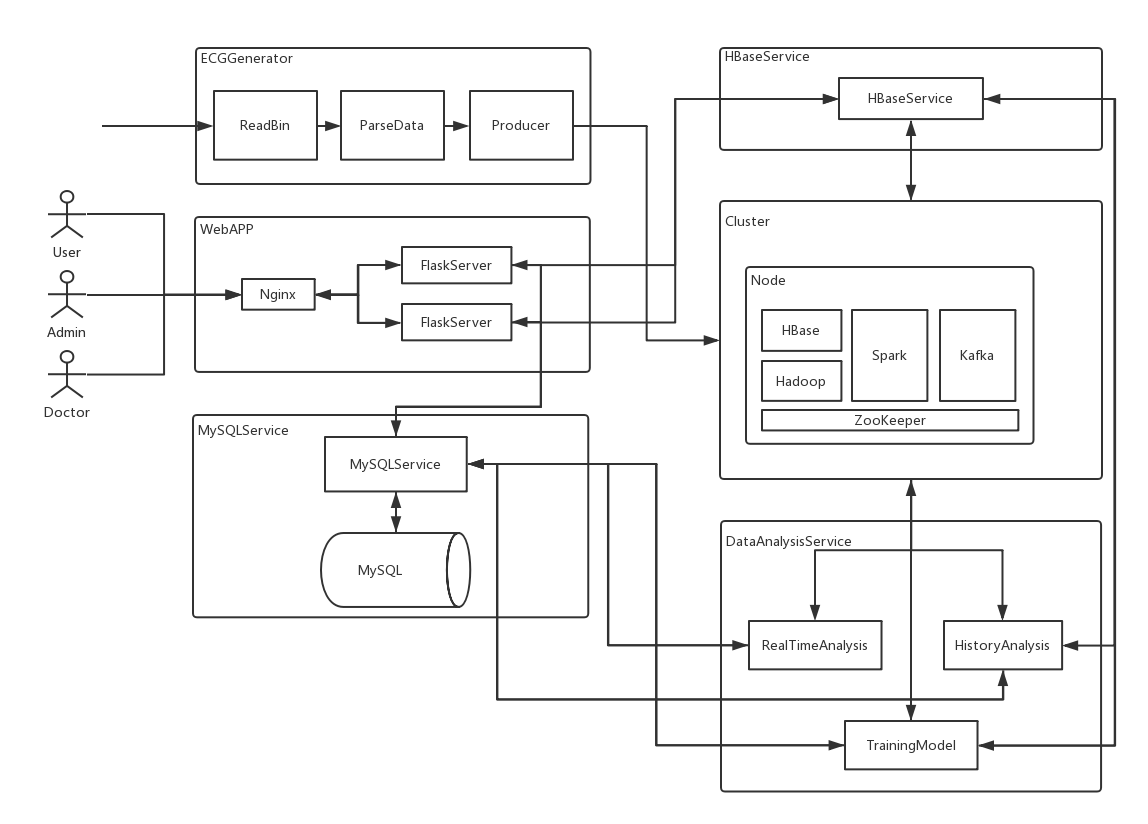
## 7.1 部署架构



首先Nginx分发业务请求给业务子系统，业务子系统接收请求，与数据库交互。如果是个人心跳历史数据（文件形式），则把它存到HBASE，然后调用Spark处理系统，在HBASE请求数据，对数据进行处理，将处理结果存入MySQL。右侧的实时心跳数据发生器将实时数据传到Kafka消息队列，然后Spark的实时心跳数据处理模块会不断的从消息队列中取出数据进行处理，然后将处理结果存入MySQL。前端结果显示，直接从MySQL读取即可。

## 7.2 逻辑架构





# 接口设计

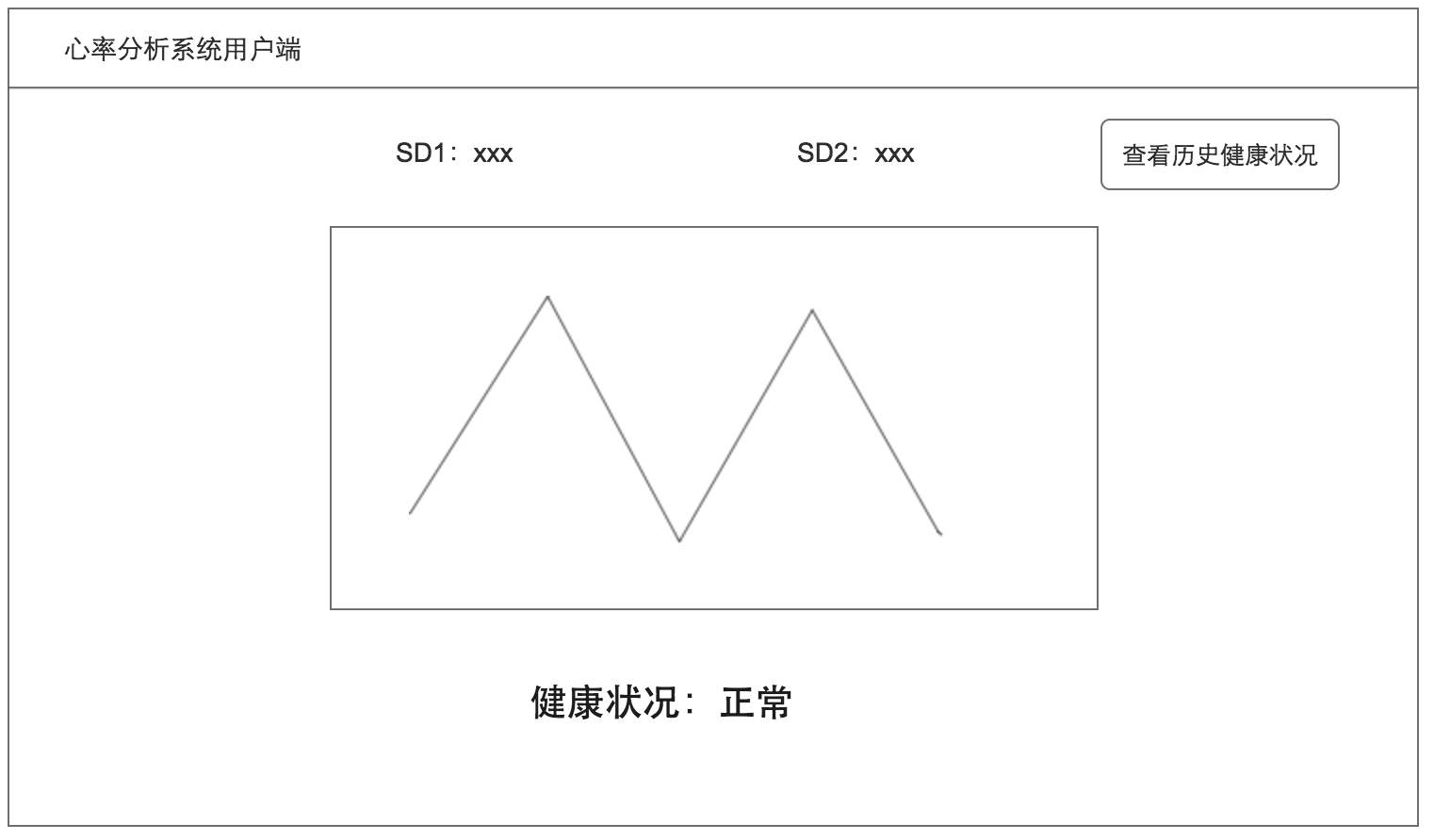
## 8.1 用户接口

* 登录注册
* 普通用户界面

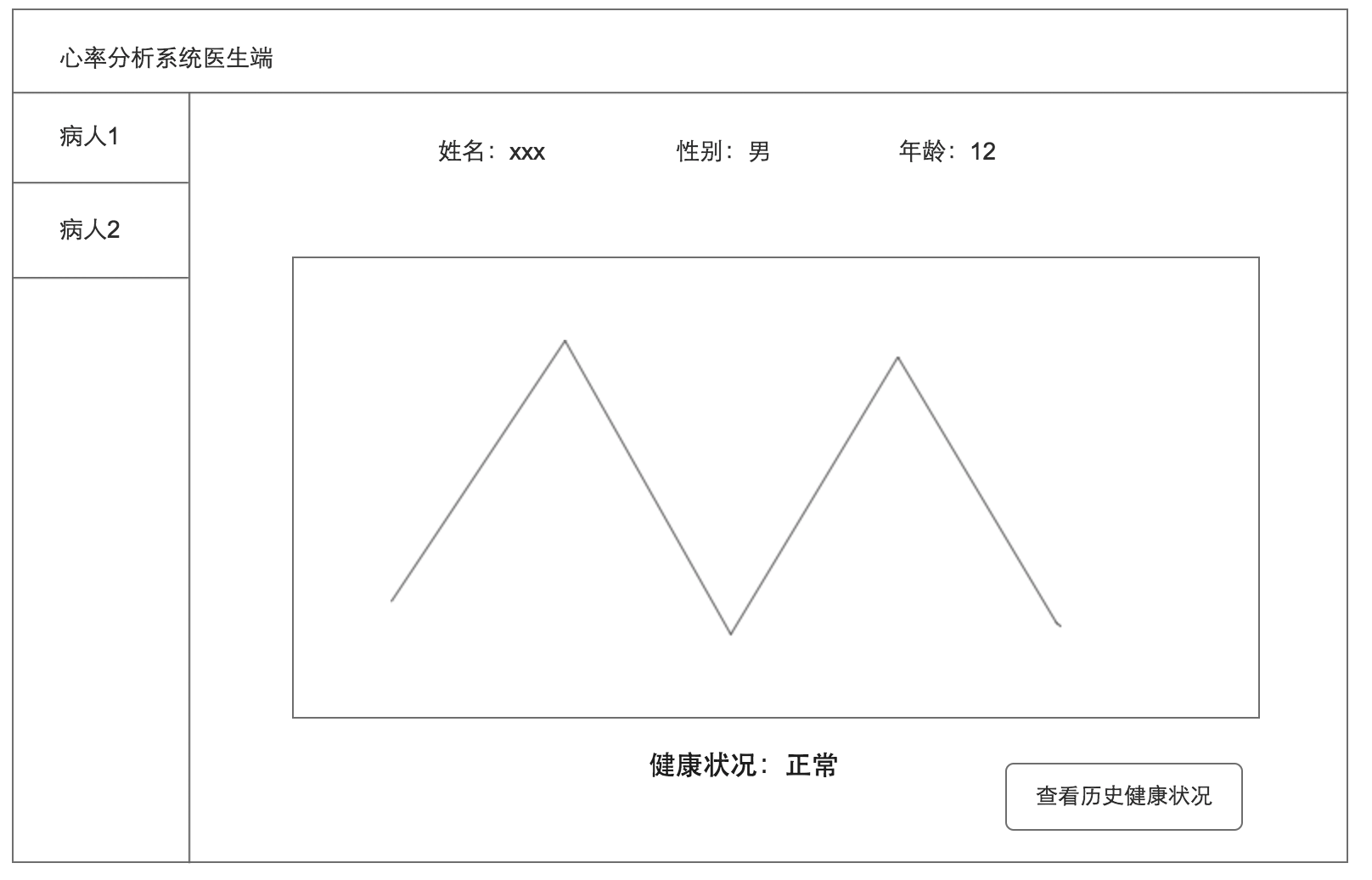
上传心率文件



健康报告



医生监控页面



管理员界面

  医生管理



病人管理



## 8.2 业务子系统

### 病人模块

**添加病人**

URL: /hbr/patient/add

参数：name, gender, position (string, 病人的床位), doctorId (指定医生的ID)

返回：参考“登录”的接口设计

描述：添加病人默认status是treating

**查询病人列表（医生端）**

URL: /hbr/patient/getlistbydoctor

参数：无

返回：{“code”: 200, “data”: [{“id”: 100, “name”: “xxx”, … (具体属性参考数据库)}]}

描述：根据当前登录的用户（医生），筛选出属于特定医生并且status等于treating的病人记录。

**查询病人列表（管理员端）**

URL: /hbr/patient/getall

参数：无

返回：{“code”: 200, “data”: [{“id”: 100, “name”: “xxx”, … (具体属性参考数据库)}]}

### 用户模块

**登录**

URL： /hbr/user/login

参数：username, password

返回： 登录成功： {“code”: 200, “msg”: “OK”}

登录失败： {“code”: -1, “msg”: “”}

**注册**

URL： /hbr/user/register

参数：username, name, password (MD5加密), gender

返回：形式同“登录”

说明：注册的用户角色默认为common，角色字段所有可能值参考数据库注释。

**添加医生**

URL： /hbr/user/adddoctor

参数：username, name, password (MD5加密), gender

返回：形式同“登录”

说明：角色后台设置为doctor，角色字段所有可能值参考数据库注释。

### 心率数据处理模块

**获取心跳数据**

URL： /hbr/heart/getfordisplay

参数：patientId

返回：{"code”: 200, “data”: {“heartBeat”: [89759, 563753,643340…], “analysisResult”: “healthy”}}

描述：查询特定patient的心跳数据及分析结果

## 8.3 Spark子系统

实时心率数据分析

历史心率数据分析

模型训练

## 8.4 Kafka消息队列

搭建系统， Restful接口

## 8.5 实时心跳发生器

病人ID，{批量心跳数据}

# 总结