# 【疫情管控系统】

## ——病例监测结果发布子系统

# 总体设计说明书

组长：王晨露 3180103485@zju.edu.cn

组员：康锦辉 刘轩铭 徐晓丹 朱一丁

日期：2020/05/24

版本：Version 2.0

# 目录

# 1 引言

## 1.1 编写目的

从本阶段开始，项目进入正式开发阶段。这份总体设计报告的编写目的，是以本项目的需求分析说明书为依据，从总体设计的角度，明确系统管理子系统的总体架构、流程、数据结构、数据库设计。目的在于：

- 为开发人员提供依据
- 为修改、测试、维护提供条件
- 明确各模块外部接口，内部接口，用户接口
- 项目负责人按计划说明书的要求布置和控制开发工作全过程

**本说明书的预期读者包括：**

- 软件客户
- 项目经理
- 测试人员
- 项目管理和开发人员
- 系统维护人员

## 1.2 项目背景

- **软件系统名称**
  - 疫情管控系统
- **任务提出者**
  - 浙江大学软件工程基础课程任课老师-张引
- **开发者**
  - 由浙江大学 2019-2020 学年夏学期软件工程基础课程部分学生组成的项目组
- **用户**
  - 买家、卖家、系统管理员
- **实现该软件的计算机网络**
  - 由若干台 PC 机组成的局域网
  - 互联网
- **相关背景介绍**

浙江大学软件工程基础课程分为理论课与实践课两个部分。在理论课中，教师有选择地介绍了与软件工程基础相关的理论，强调并确定了适用于整个软件生命期的基本原则，全面而深入地介绍了这些基本原则在软件设计、规范、验证、软件生产过程和管理 活动中的运用。而实验课采取分组形式完成，每 5 个学生为一组，分别设有组长、主程序员、程序员、测试员、文档员等角色。本次课程，教师选取疫情管控系统作为综合性实验题目本次课程。其中我组主要负责病例监测结果发布子系统，实现对病例结果的计算和监控功能，主要面向用户是关注新冠肺炎疫情情况的社会各界人士。

## 1.3 相关定义

(1) MySQL

一个小型关系型数据库管理系统。

(2) JavaScript

JavaScript 是一种面向对象的动态类型的区分大小写的客户端脚本语言。

(3) React

React 是一个用于构建用户界面的 JavaScript 库。它拥有较高的性能，代码逻辑非常简单，被越来越广泛地使用。

(4) SQL 注入

通过把 SQL 命令插入到 Web 表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的 SQL 命令。

(5) 数据库连接池

数据库连接池负责分配、管理和释放数据库连接，它允许应用程序重复使用一个现有的数据库连接，这项技术能明显提高对数据库操作的性能。

(6) UTF-8

UTF-8 是 UNICODE 的一种变长字符编码，又称万国码安全证书。安全证书是在进行网上交易时的身份证，或者说是私人钥匙，安全证书是唯一的，与任何其他人的证书都不相同。

(7) MD5 加密算法

Message Digest Algorithm MD5（消息摘要算法第五版）为计算机安全领域广泛使用的一种散列函数，用以提供消息的完整性保护。

## 1.4 系统概述

疫情管控系统是用于快速发布疫情相关权威信息的现代化管理系统。各管理员可通过各自的管理员账户在相应子系统中实时发布相关信息，而用户则可在 Web 端实时查看疫情动态变化信息从而实现对疫情的快速管控和防控知识宣传。

该系统由六个模块构成，分别为用户管理子系统、疫情新闻发布子系统、病例监测结果发布子系统、同乘交通自查子系统、复工及人口流动子系统以及物资申领子系统。具体模块的功能要求如下：

**（一） 用户管理子系统与疫情新闻发布子系统：**

用户管理子系统允许超级管理员根据管理员的个人信息建立其余五个子系统的管理员账户。允许普通游客使用个人信息（如真实姓名，身份证号码和电子邮件地址）注册个人账户，并随时修改此信息以及个人密码。此外允许用户以游客身份进入。超级管理员的权限为建立其余四个子系统管理员账户，管理员账户权限为管理各自的子系统，游客可查看其余四个子系统的前端界面信息，个人用户可在疫情新闻发布子系统发表新闻评论，以及在物资申领子系统中申领物资。疫情新闻发布子系统允许新闻

管理员用户在后台实时发布最新的疫情新闻。新闻分为辟谣新闻、方法知识、抗疫进展三类，在前端界面按时间顺序进行排列，个人用户可在前端查看新闻具体内容并可对新闻内容进行评价、分享，而游客不能发表评论。

**（二） 病例监测结果发布子系统**

允许各省管理员用户在后台发布各省每日确诊、境外输入、无证感染、治愈、死亡人数。并通过累计计算得出全国现存确诊总人数、境外输入总人数、现存无证感染总人数、累计确诊人数、累计死亡人数、累计治愈人数。而前端除呈现出上述数据外还应根据上述信息形成国内疫情地图，全国治愈率/死亡率趋势图，国内湖北以及非湖北地区疫情各类人数(死亡、确诊、治愈)趋势图，湖北/非湖北地区新增确诊趋势图以及各省境外输入对比图。

**（三） 同乘交通自查子系统**

允许同乘自查子系统管理员在后台发布高危列车、航班信息（编号、起点、终点、时间）。个人用户在填写个人具体信息（真实姓名、身份证号、手机号、地址、邮箱）后可进行同乘自查。前端显示所有高危班次且支持相关信息的模糊检索。在匹配的班次后提供登记入口，用户进入后可登记个人信息并提醒居家隔离。管理员在后台可导出各班次的用户登记信息，且支持后台邮件群发功能，从而达到快速通知用户的目的。

**（四） 复工及人口流动子系统**

允许各省的该板块管理员在后台发布各省市的复工情况，其中包括复工复产举措、社区管控举措、交通出行举措、医疗服务举措。同时可在后台发布其余各省人口流入本省的人口数量。在前端界面除展示工复产举措、社区管控举措、交通出行举措、医疗服务举措外，还应展示全国人口流动图。

**（五） 物资申领子系统**

个人用户可在此系统前端填写个人信息（真实姓名、身份证号、手机号、地址、邮箱）申领各类政府发放物资。该子系统管理员可在后台发布各类物资的申领入口，且对各类物资进行截止时间设定。到达截止时间将自动关闭申领入口，系统随机自动摇号将结果自动发邮件通知各参与人。系统管理员在后台可导出中签人的个人信息列表。

## 1.5 项目概述

病例监测结果发布子系统是疫情管控系统的模块之一。它允许各省管理员用户在后台发布各省每日确诊、境外输入、无证感染、治愈、死亡人数。并通过累计计算得出全国现存确诊总人数、境外输入总人数、现存无证感染总人数、累计确诊人数、累计死亡人数、累计治愈人数。而前端除呈现出上述数据外还应根据上述信息形成国内疫情地图，全国治愈率/死亡率趋势图，国内湖北以及非湖北地区疫情各类人数(死亡、确诊、治愈)趋势图，湖北/非湖北地区新增确诊趋势图以及各省境外输入对比图。

病例监测结果发布在整个系统中是一个核心模块，负责病例数量的统计，计算以及发布。

本模块设置用户和管理员两种身份，管理员用户发布和修改数据的权限，而普通用户只拥有查看已有数据的权限。项目应该严格区分两者的权限，保证信息的安全和权限的唯一性。

# 2. 总体设计

## 2.1 需求规定

### 2.1.1 系统功能

根据需求，本项目需要提供疫情数据动态显示、数据修改和获取 API 的配套前端。服务器后端，本项目需设计保存全国各省疫情动态数据和历史数据的数据库，以及一个对数据进行累加处理的计算模块。API 方面，本项目主要提供身份验证、管理员表单提交、疫情当前数据获取、疫情历史数据查询等 4 个 API。与 API 相对应，前端界面需要发布数据表单，全国疫情数据显示，以及各子数据显示等页面。

### 2.1.2 系统性能

- 界面设计应简洁直观，布局合理，清晰地呈现信息，突出重点内容。操作方便，用户容易上手。
- 系统具有良好的反应速度，给用户良好的使用体验。我们要求在良好的网络情况下，系统应具有以下时间特性要求：

  (1) 单个用户在线时 Web 响应用户动作时间小于 1 秒。数据发布操作响应用户动作时间小于 2 秒。

  (2) 500 个用户同时在线时 Web 响应用户动作时间小于 2 秒。数据发布操作响应用户动作时间小于 5 秒。
- **访问容量**：该系统至少在同一时间内支持 500 个用户并发访问。
- **服务器配置最低要求**：CPU2.6G，内存 2.0G，硬盘 7200 转。
- **数据处理能力**：至少支持 10000 条数据发布记录。
- **可用性**：该子系统应实现在大多数流行的 Web 浏览器中正确显示和执行，包括 Firefox、Chrome、Edge、IE 等。

### 2.1.3 输入输出要求

客户端通过网页展现给用户一个友好的界面，用户可以通过提交表单或者点击超链接向 服务器提供数据与命令。服务器后台处理后将结果显示到用户的网页界面上。API 则为其他子系统和前端提供清晰、简洁的接口，子系统通过 API 向服务器发送请求，服务器后台处理后返回格式化的结果；若子系统进行非法操作，服务器能够进行判断并返回错误信息，避免发送的请求影响后端稳定性

### 2.1.4 数据管理能力要求

#### 2.1.4.1 安全

**保密性**

1. 用于身份验证的用户名和密码应防止未经授权的用户访问系统。应构建访问控制以防止合法用户非法使用系统资源。
2. 某些敏感数据（如用户名和密码）在交换时应加密。密码在存储之前应加密。
3. 在用户登录期间，应该防止 SQL 注入，密码强制破解和伪造会话入侵。

**完整性**

1. 防止非法用户对数据进行无意或恶意的修改、插入、删除，防止数据丢失。
2. 防止内部用户对数据进行无意或恶意的修改、插入、删除，防止数据丢失。
3. 为数据库加上一定的约束，对关键性操作如删除、修改进行限制，并对用户进行警示。

4. 定期备份数据

#### 2.1.4.2 性能

对于频繁访问数据库的操作，后台需要建立持久的数据库连接，以避免重复连接数据库耗费资源。

## 2.2 运行环境规定

由于实验条件有限，我们并不能提供专门的服务器运行系统，故将利用配置较高的 PC 作为服务器，保证服务器以及客户端间网络畅通即可。

**计算机：**

- **CPU**：不小于 2.0GHz
- **内存**：不小于 2.0GB

**通讯设备：**

- **网线**：有良好数据传输能力
- **网卡**：100M

**软件依赖：**

- **操作系统**：Windows Vista/7/8/8.1/10, Mac OS，Linux
- **数据库平台**：MySQL
- **前端开发框架**：React
- **后端开发框架**：Django
- **Web 服务器**：Apache
- **MySQL 管理软件**： MySQL WorkBench 等
- **开发工具**：能支持网页开发的工具均可（如 IDEA）
- **测试工具**：能支持测试的工具均可（如 JEST）
- **建模工具**：Microsoft word
- **办公软件**：Microsoft Office
- **浏览器**：Chrome、Edge、Safari（移动端）

## 2.3 基本设计概念和处理流程

本子系统是一个横跨前后端的大模块，主要负责安全身份验证、表单信息提交、疫情数据可视化等子模块。

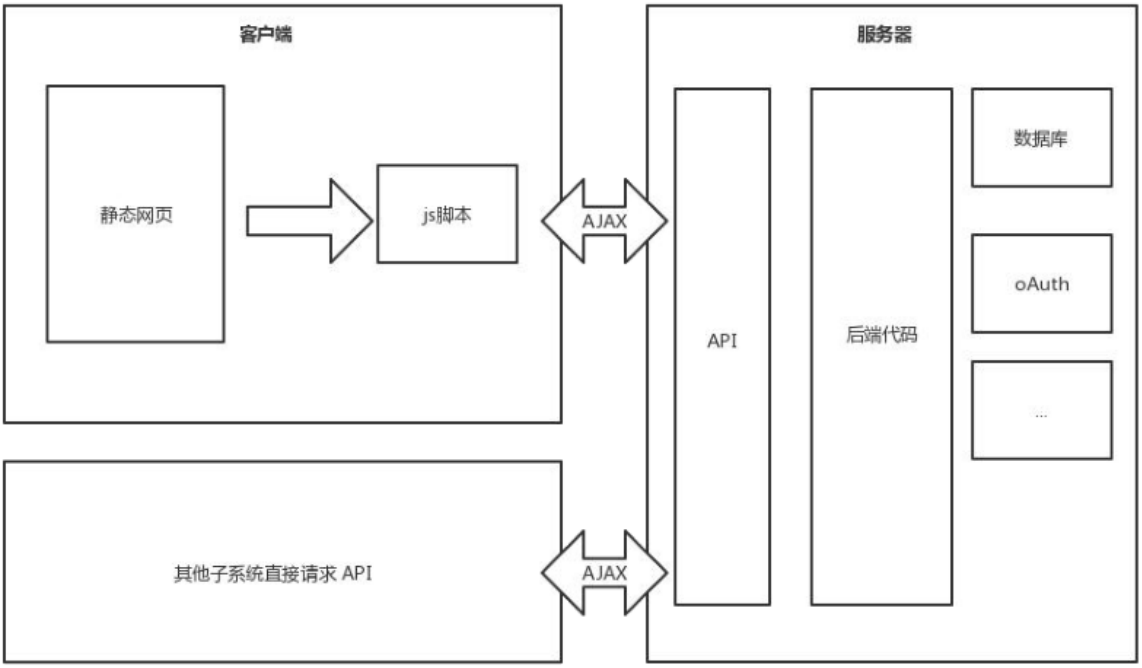
**前端：**基于 JavaScript 的 React 框架

**后端：**基于 Python 的 Django 框架

**数据库：**数据库采用 MySQL

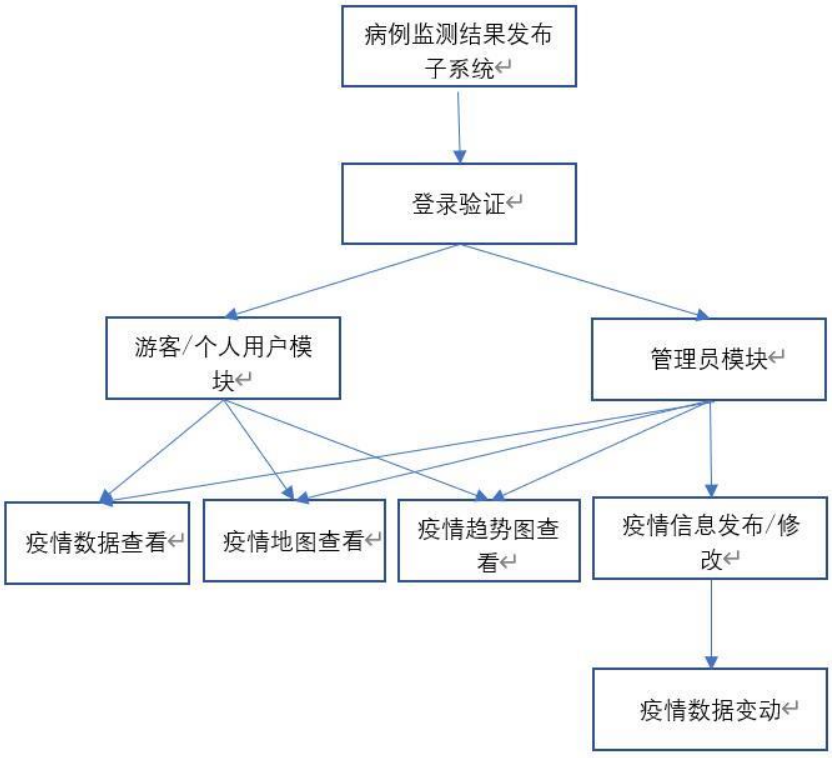
**客户端：**考虑到本系统所使用的框架较新，推荐使用 Chrome、Firefox、Safari 或 Microsoft Edge 浏览器。不推荐使用任何一个版本的 IE 浏览器，如需使用，版本须在 11 及以上。

**处理流程图如下：**



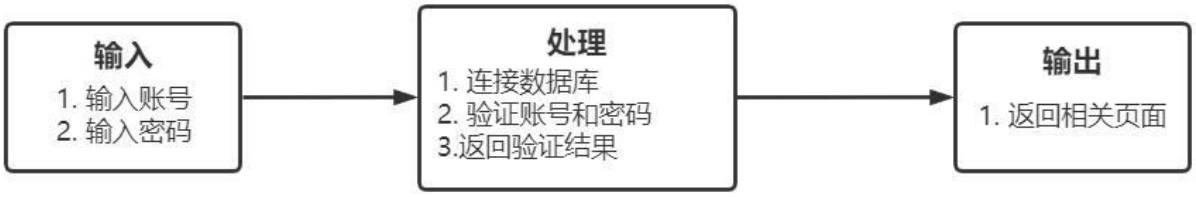

## 2.4 结构

### 2.4.1 病例监测结果发布系统层次图

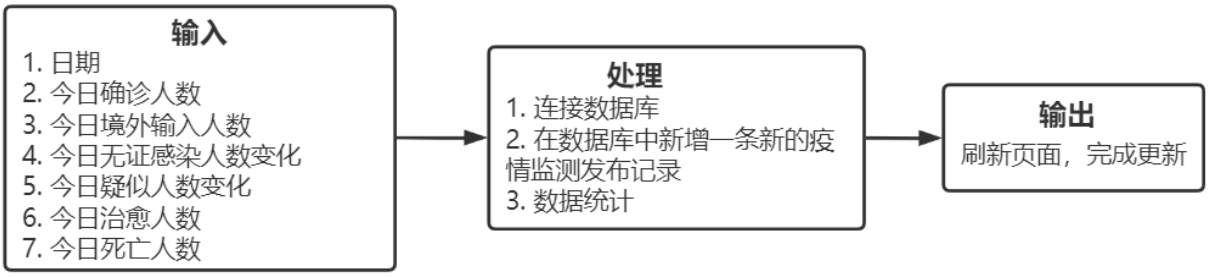


## 2.4.2 功能 IPO 图

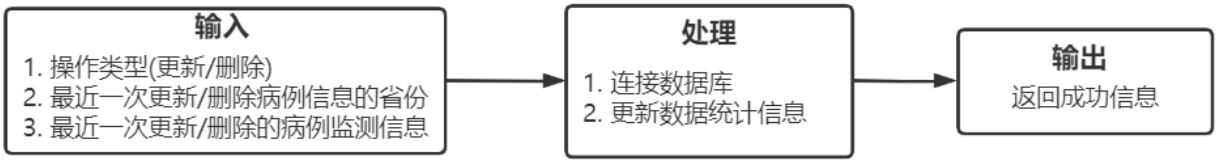
### 2.4.2.1 登录验证



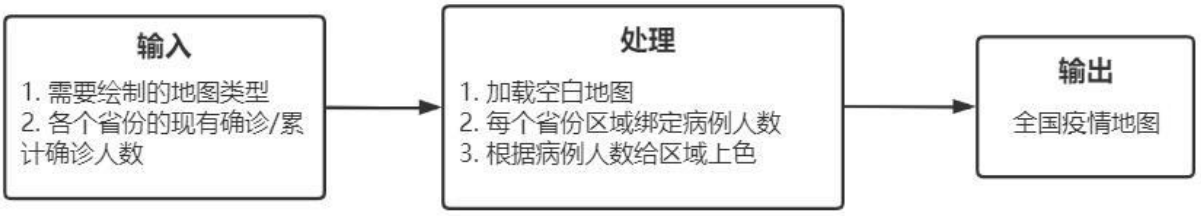

### 2.4.2.2 数据发布



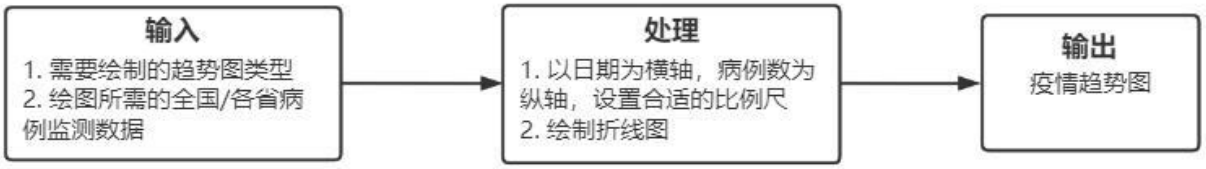

### 2.4.2.3 数据统计



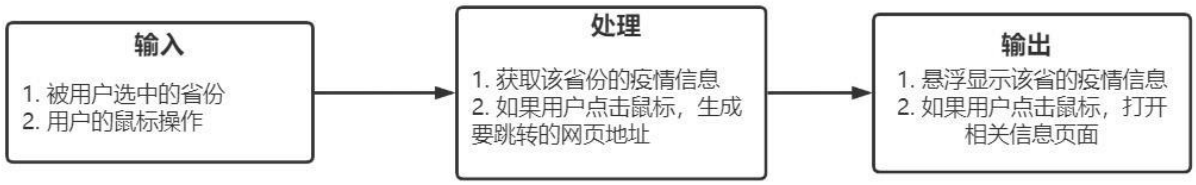

### 2.4.2.4 生成疫情地图



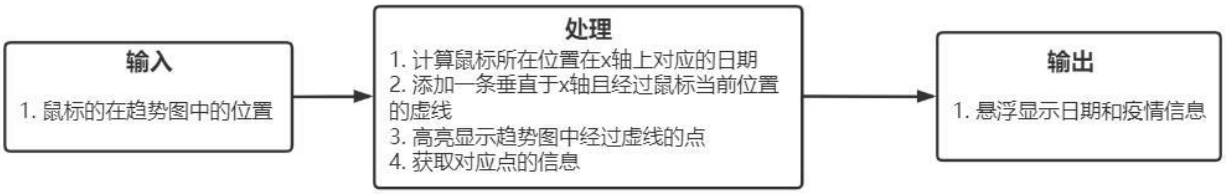

### 2.4.2.5 生成疫情趋势图



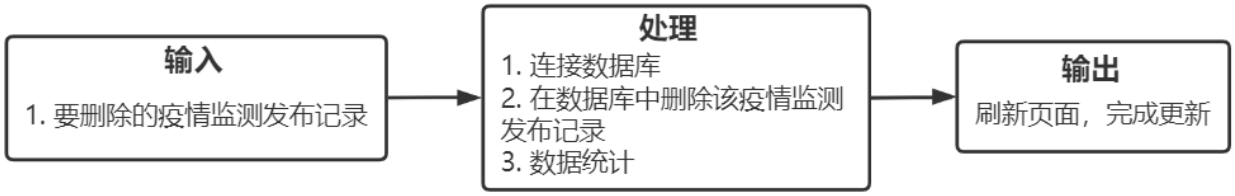
#### 2.4.2.6 查看疫情地图




#### 2.4.2.7 查看疫情趋势图



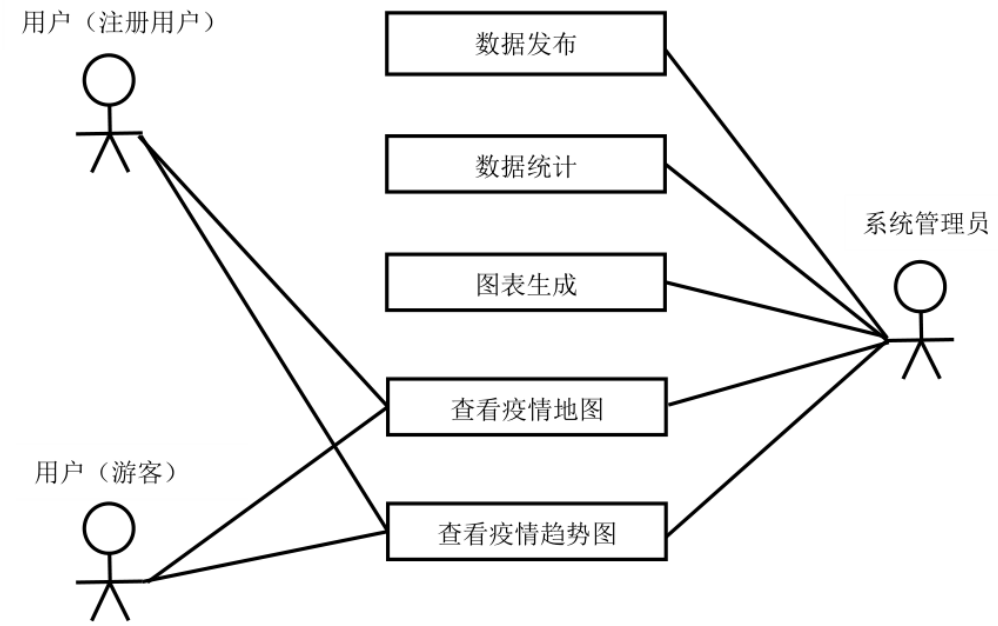

#### 2.4.2.8 数据删除



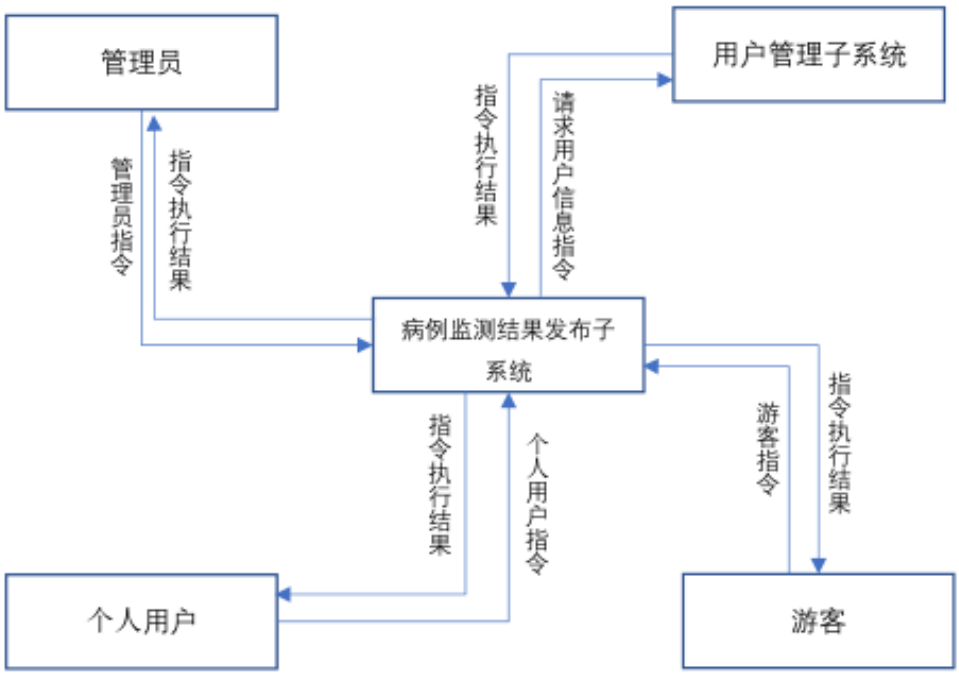

## 2.5 系统结构

### 2.5.1 系统功能结构

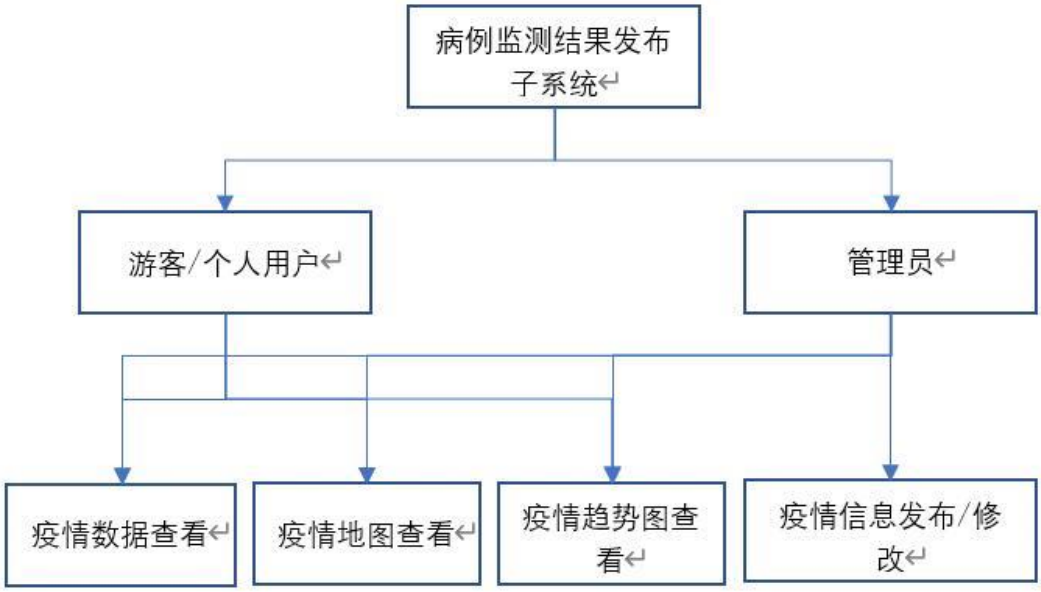
通过需求分析，已经完成对系统的用例分析，具体如下所示：



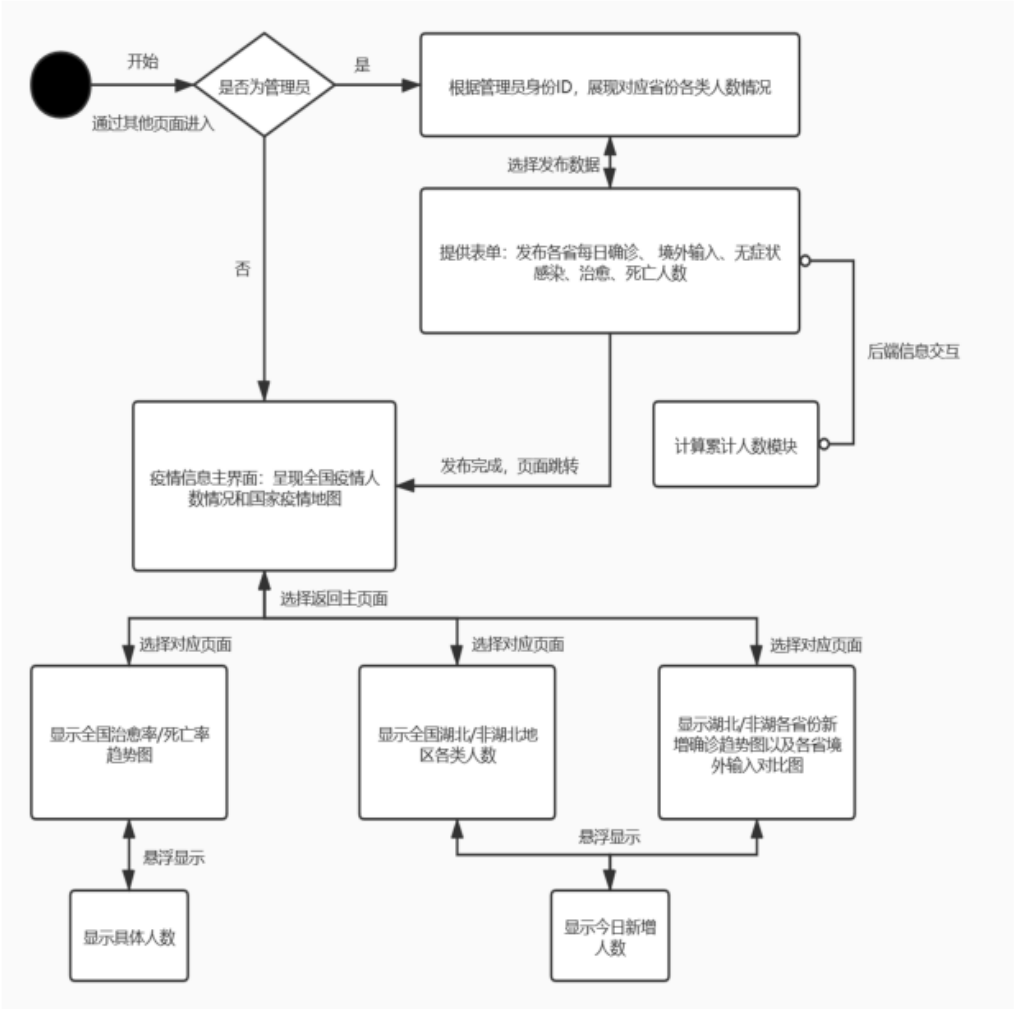

需求分析中，已经完成了对数据流相应的分析，其顶层数据流分析具体如下图所示：、




通过系统需求的分析，系统的总体结构设计变得明晰了。系统的系统层次图如下所示：




需求分析中完成了系统的状态图设计，这里引用状态图作为系统运行的整体流程参考。具体如下图所示：



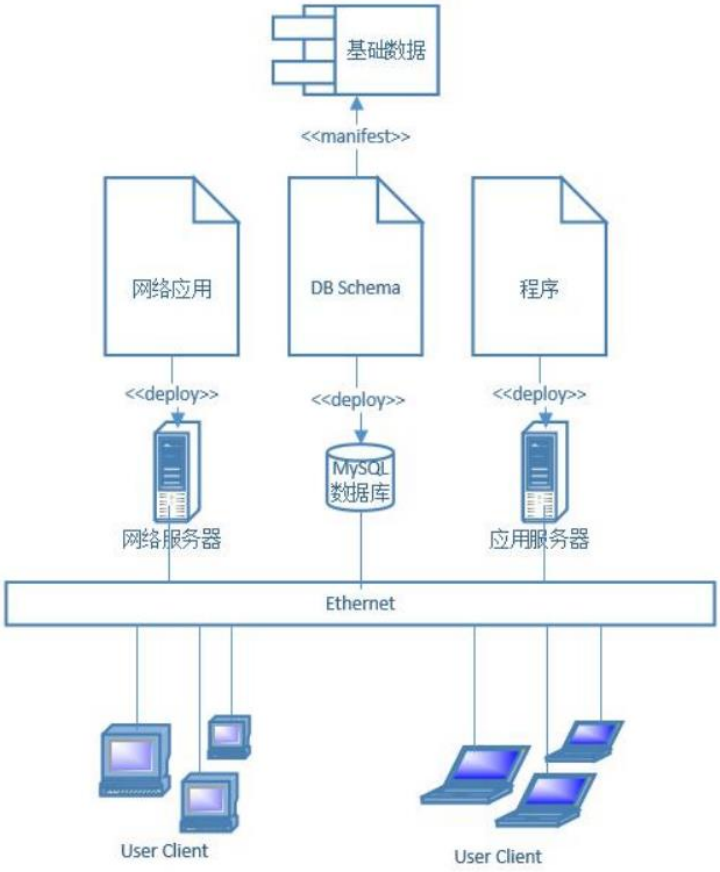
### 2.5.2 技术简介

在系统设计中，我们将采用基于 Python 的 Django 框架来进行 Web 页面的构建。 Django 是一个开放源代码的 Web 应用框架，由 Python 写成在前端部分。使用 Django，只要很少的代码，Python 的程序开发人员就可以轻松地完成一个正式网站所需要的大部分内容。它的特点是带有强大的数据库功能，自带强大的后台功能以及具有优雅的网址。

我们采用 JavaScript 的 React 框架进行装饰。React 使创建交互式 UI 变得轻而易举。为应用的每一个状态设计简洁的视图，当数据改变时 React 能有效地更新并正确地渲染组件。同时 React 以声明式编写 UI，可以代码更加可靠，且方便调试。

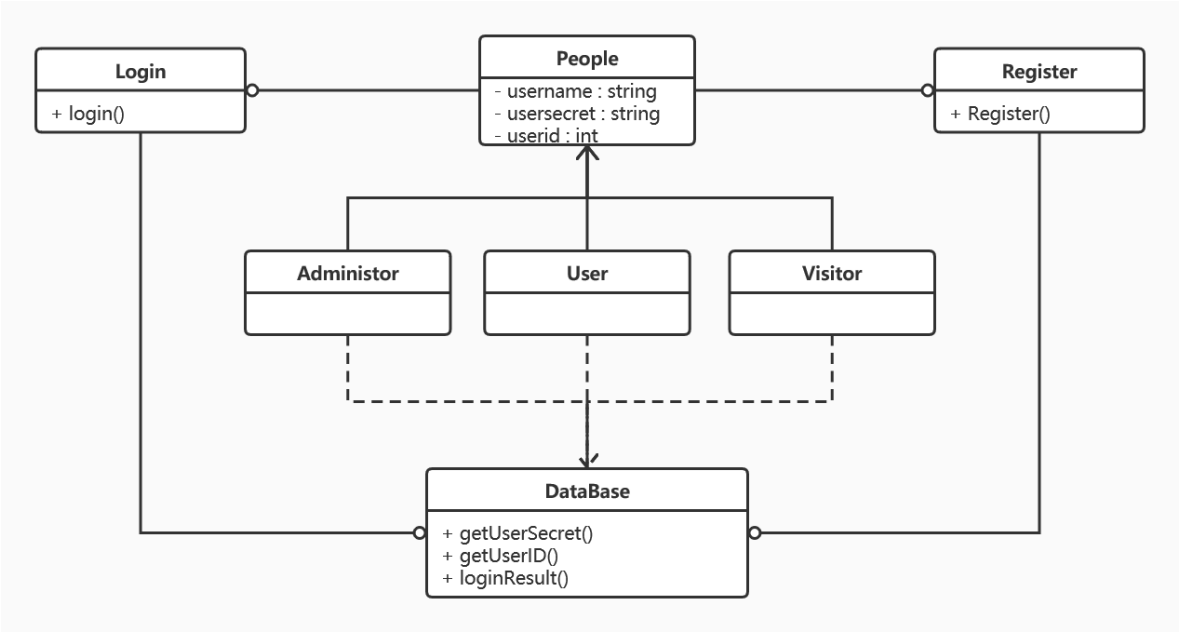
此外，我们将采用 Model-View-Controller(MVC, 三层架构模式)模型对整个项目进行设计和实现。MVC 模式是软件工程中的一种软件架构模式，把软件系统分为三个基本部分：模型(Model)、视图(View)和控制器(Controller)。在这种模式下，程序的开发人员能够将模型、视图、控制器这三个模块独立开来，从而能够对各个模块进行更为方便的修改和完善。在这个模式中，“模型” 不依赖“视图”和“控制器”，它用于封装与应用程序的业务逻辑相关的数据以及对数据的处理方法，能够对数据库中的数据进行直接访问。模型中数据的变化一般会通过一种刷新机制被公布，而为了实现该机制，相应的视图必须事先注册，以了解相应的数据改变。“视图”的任务则是实现数据有目的的显示。它需要访问相应的数据模型从而实现上面我们提到过的刷新功能。“控制器”则负责处理事件并做出响应。

### 2.5.3 部署图

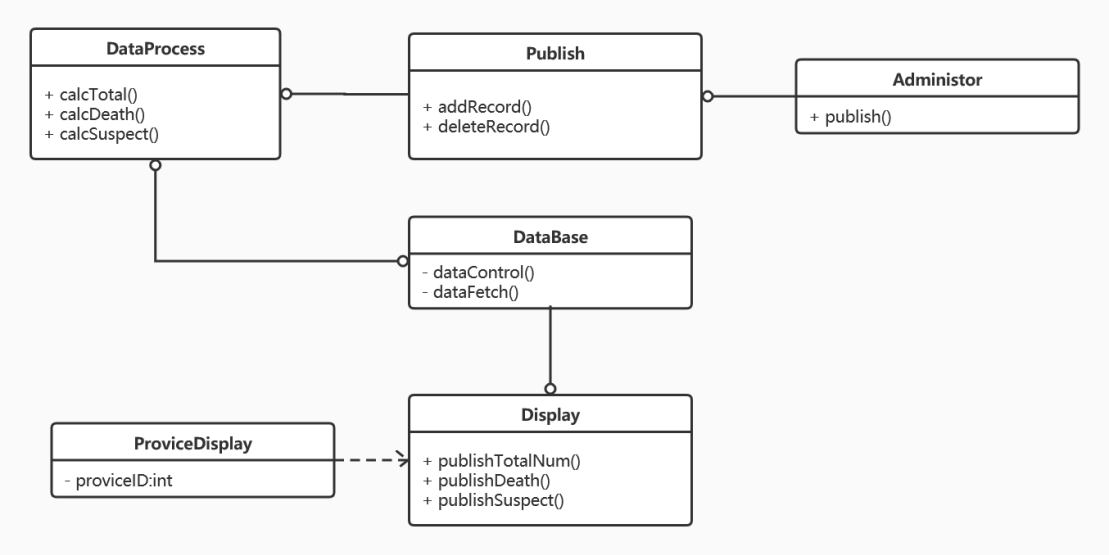


### 2.5.4 类图

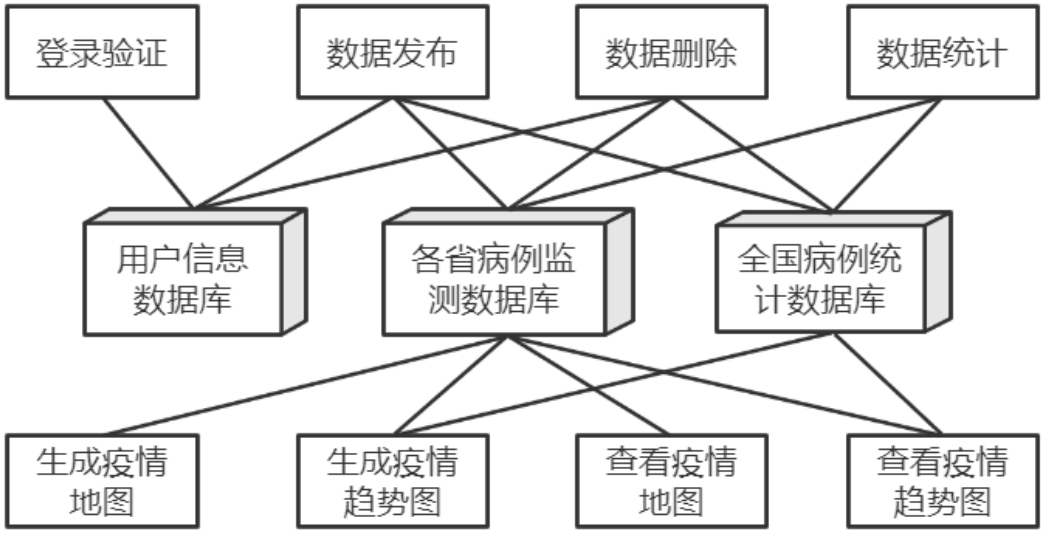
登录和注册部分：




数据处理和展示部分：

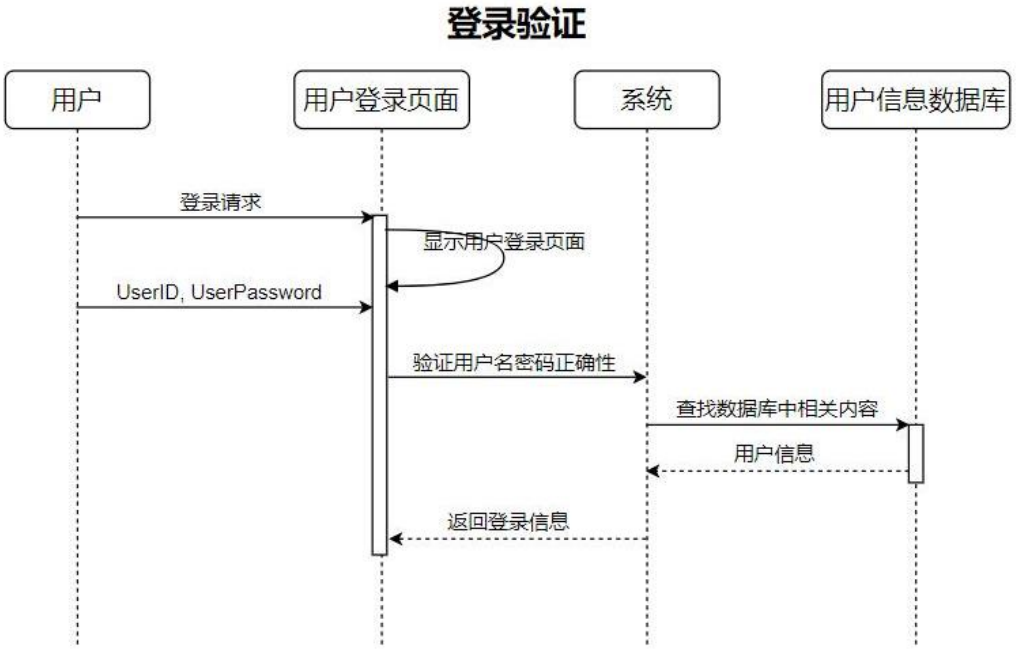


### 2.5.5 内部接口图

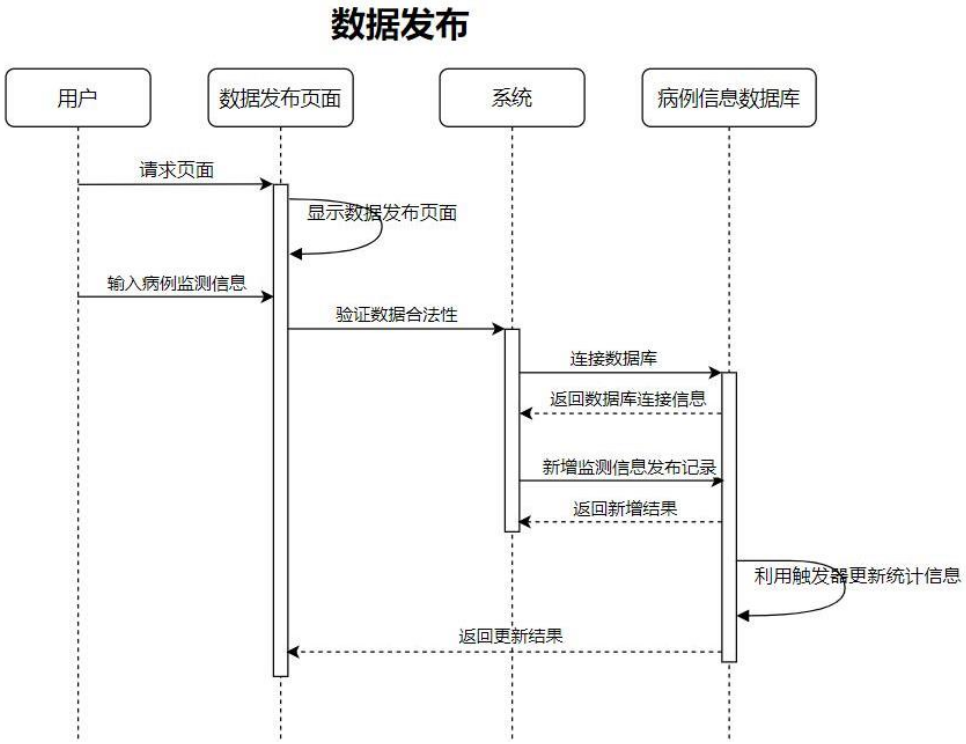



### 2.5.6 顺序图

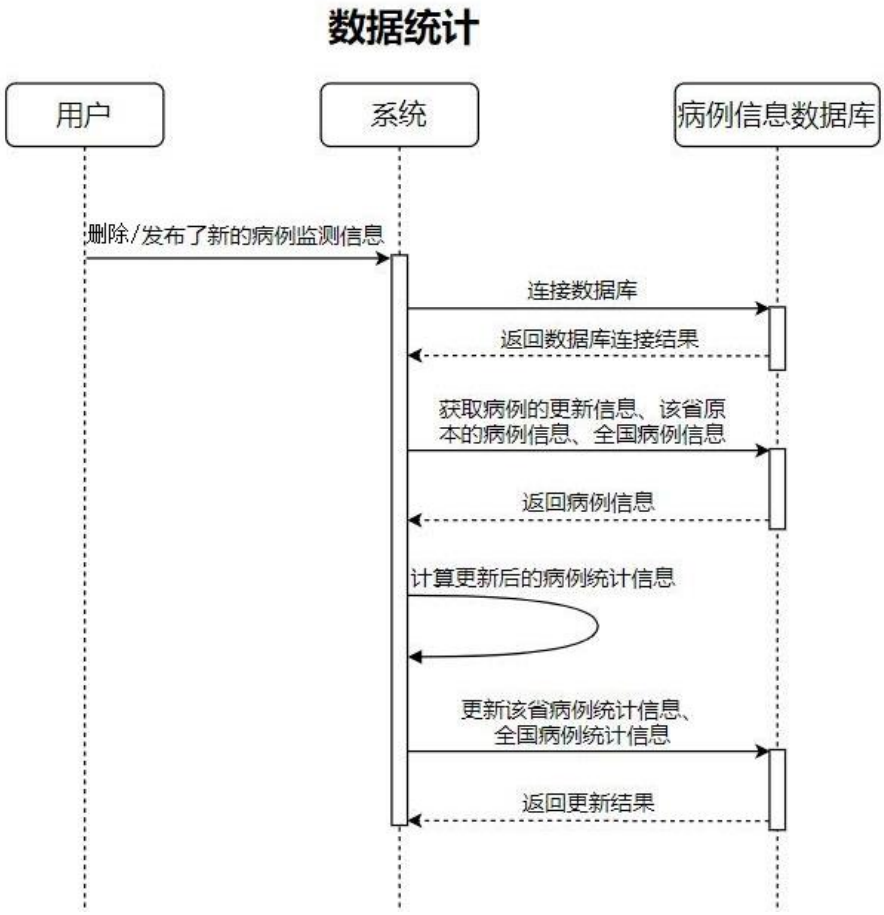
#### 2.5.6.1 登录验证



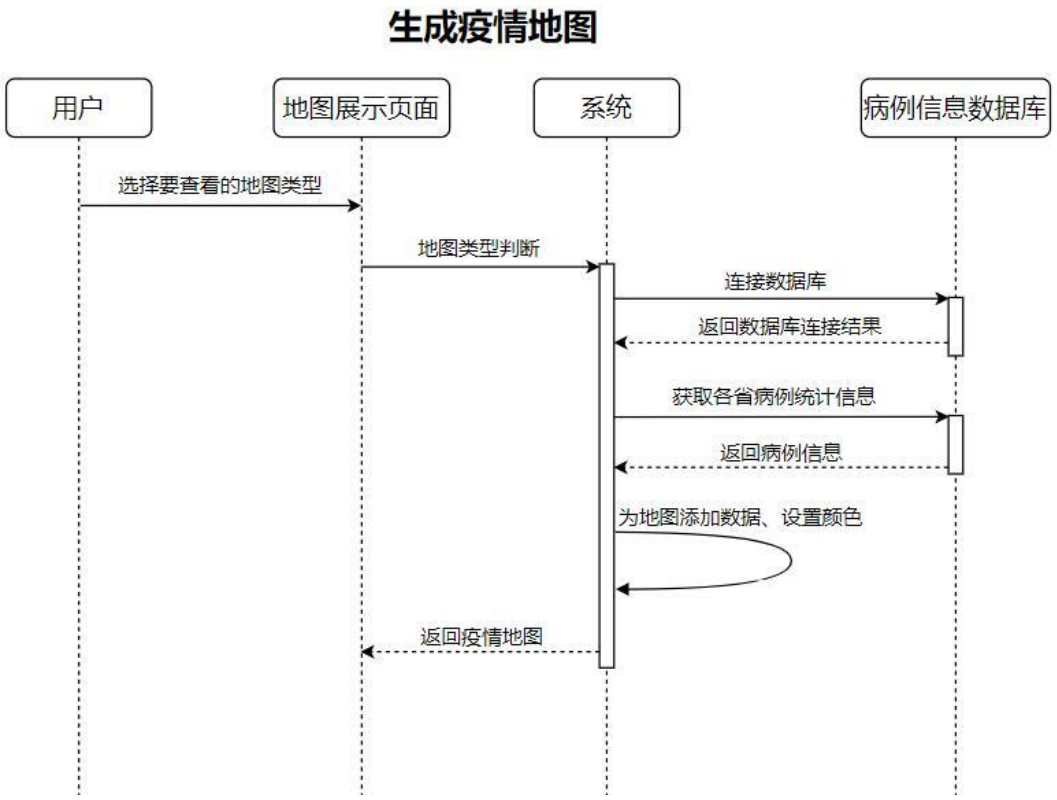
### 2.5.6.2 数据发布



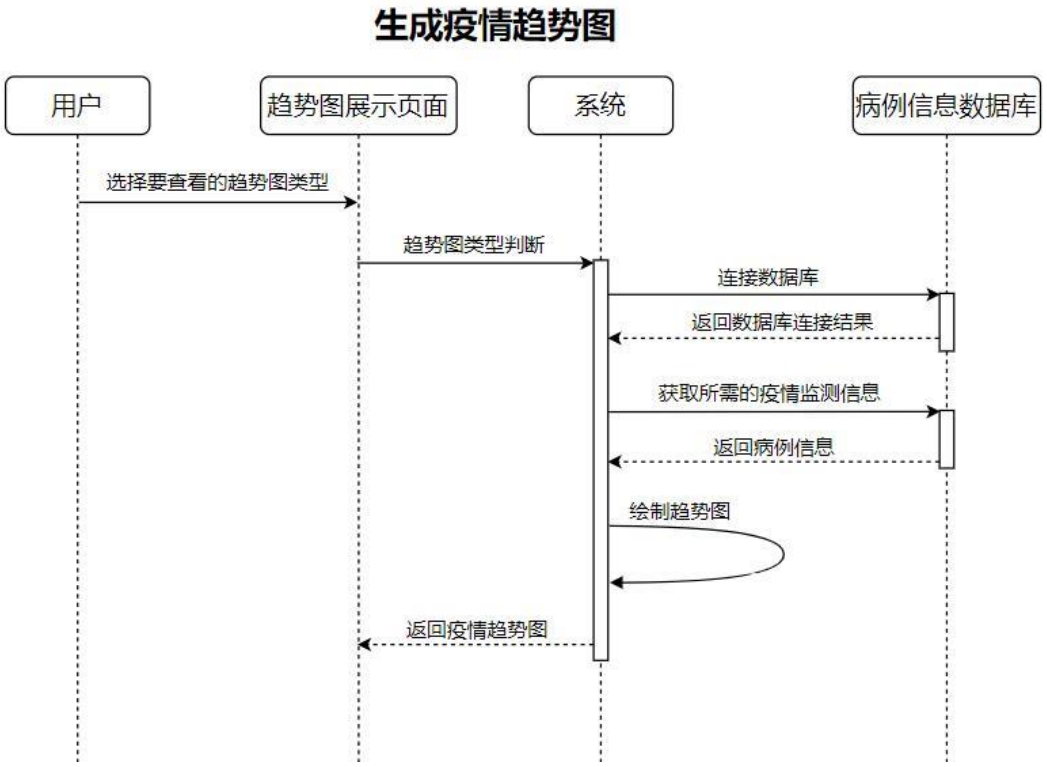

### 2.5.6.3 数据统计



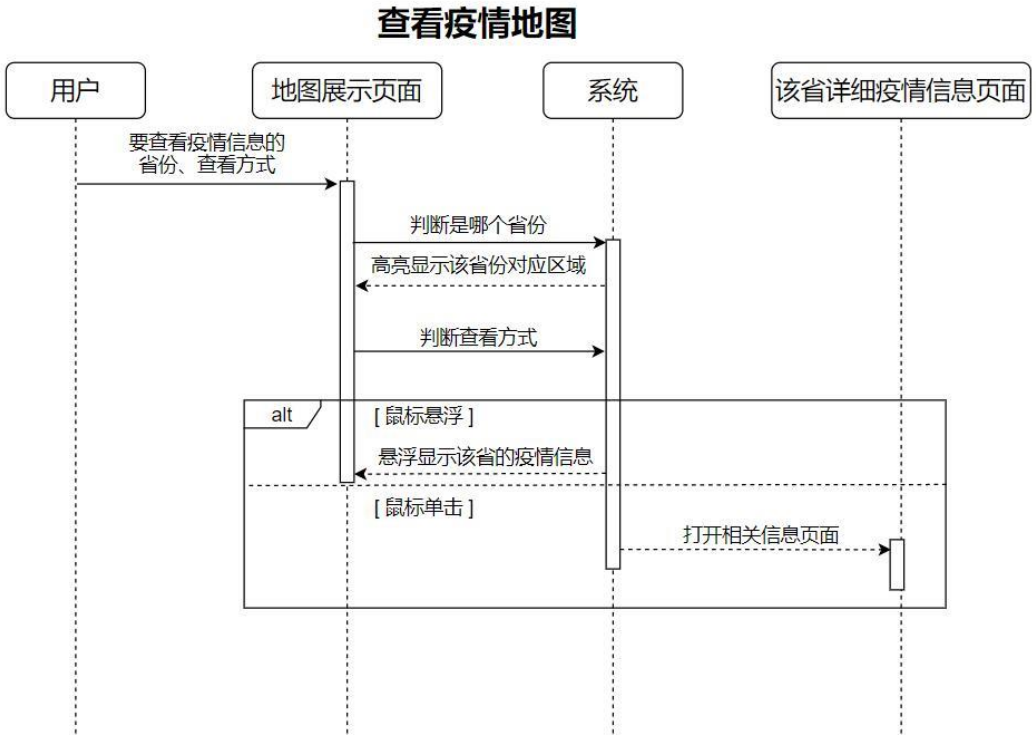
#### 2.5.6.4 生成疫情地图



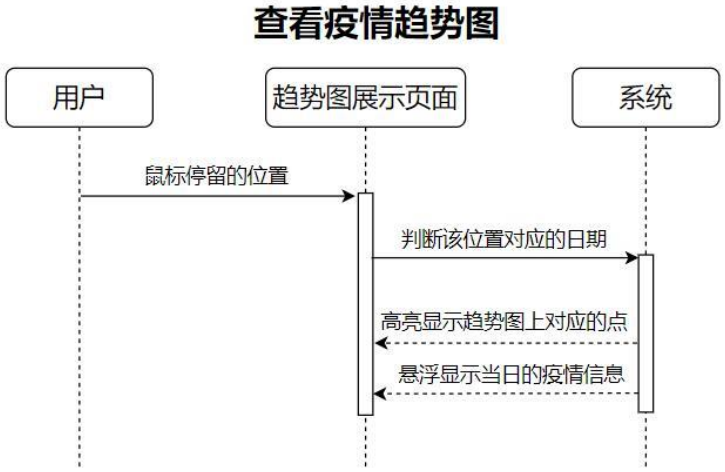

#### 2.5.6.5 生成疫情趋势图



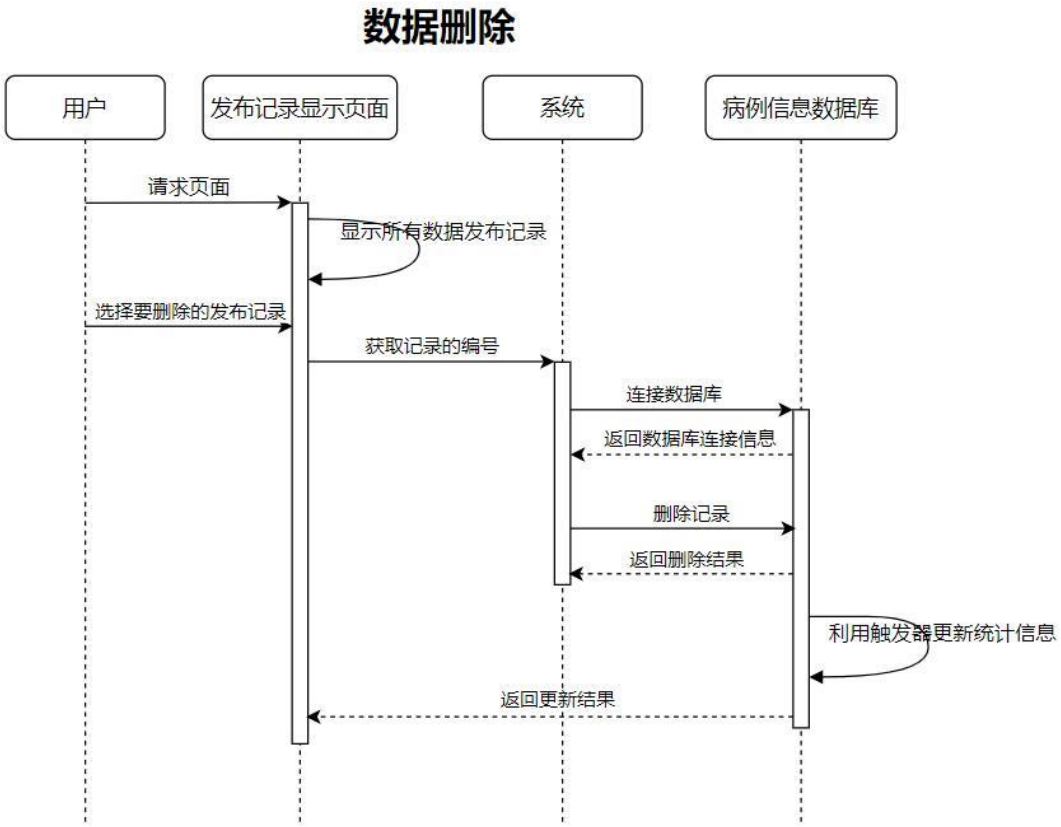
### 2.5.6.6 查看疫情地图




### 2.5.6.7 查看疫情趋势图



#### 2.5.6.8 数据删除



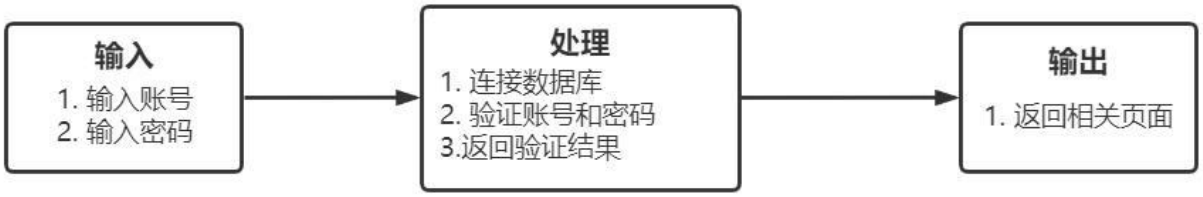

# 3 详细设计

## 3.1 登录验证

### 3.1.1 模块概述

本模块为用户（注册用户/各省管理员）登录系统时展示的页面，在此界面进行身份认证成功后才能获得相应的操作权限。存在的目的是确认用户（注册用户/各省管理员）的身份，从而判断用户是否有权限使用部分功能，保证系统安全性。

### 3.1.2 IPO 图




### 3.1.3 功能

输入账号密码后登陆，继而进行身份认证。

### 3.1.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 用户账号 | UserID | Varchar(10) | 外部输入 |
| 用户密码 | UserPassword | Varchar(20) | 外部输入 |

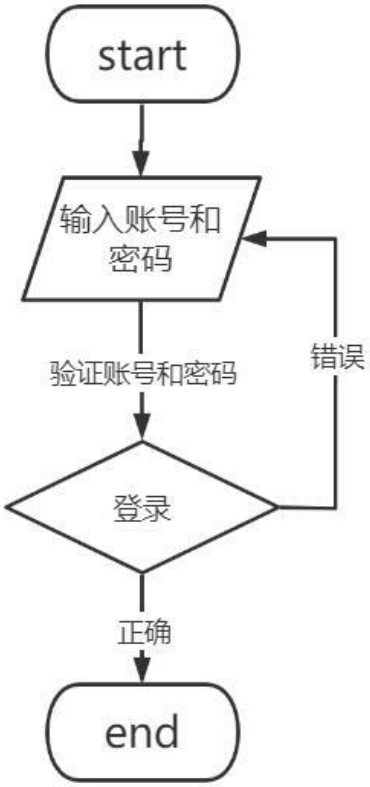
### 3.1.5 输出项

| 名称 | 标识 | 类型和格式 | 输出方式 |
| --- | --- | --- | --- |
| 登录结果 | LoginResult | Bool | 由脚本输出 |

### 3.1.6 设计方法（算法）

```
if 用户账号未登录
    检验环境安全性
    创建数据库连接
    检测输入信息是否合法（如信息长度是否足够等）
    if 信息合法
        存储信息
        查询数据库中相关信息
        进行比对验证
        if 比对成功
            返回成功信息（登陆成功）
        else
            返回失败信息（密码错误）
    else
        返回失败信息（信息非法）
else 返回用户详细信息界面
```

### 3.1.7 流程图




### 3.1.8 测试计划

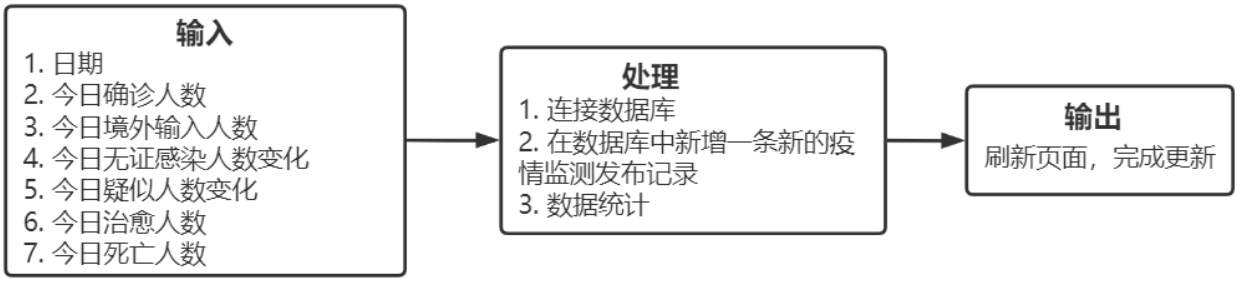
| 输入数据 | 预期结果 |
| --- | --- |
| 输入不合适的信息 | 返回错误信息，输入信息不合法 |
| 输入不匹配的账号和密码 | 返回错误信息，密码错误 |
| 输入匹配的账号和密码 | 返回成功信息，登录成功 |

## 3.2 数据发布

### 3.2.1 模块概述

在这个模块中，各省的系统管理员可以在后台发布各省每日确诊、境外输入、无证感染、疑似、治愈、死亡人数。

### 3.2.2 IPO 图




### 3.2.3 功能

更新各省病例检测情况。

### 3.2.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
|---|---|---|---|
| 日期 | Date | Varchar(10) | 外部输入->脚本转换 |
| 今日确诊人数 | ConfirmedNumber | Int | 外部输入 |
| 今日境外输入人数 | ImportedNumber | Int | 外部输入 |
| 今日无证感染人数变化 | AsymptomaticNumber | Int | 外部输入 |
| 今日疑似人数变化 | SuspectedNumber | Int | 外部输入 |
| 今日治愈人数 | CuredNumber | Int | 外部输入 |
| 今日死亡人数 | DeathNumber | Int | 外部输入 |

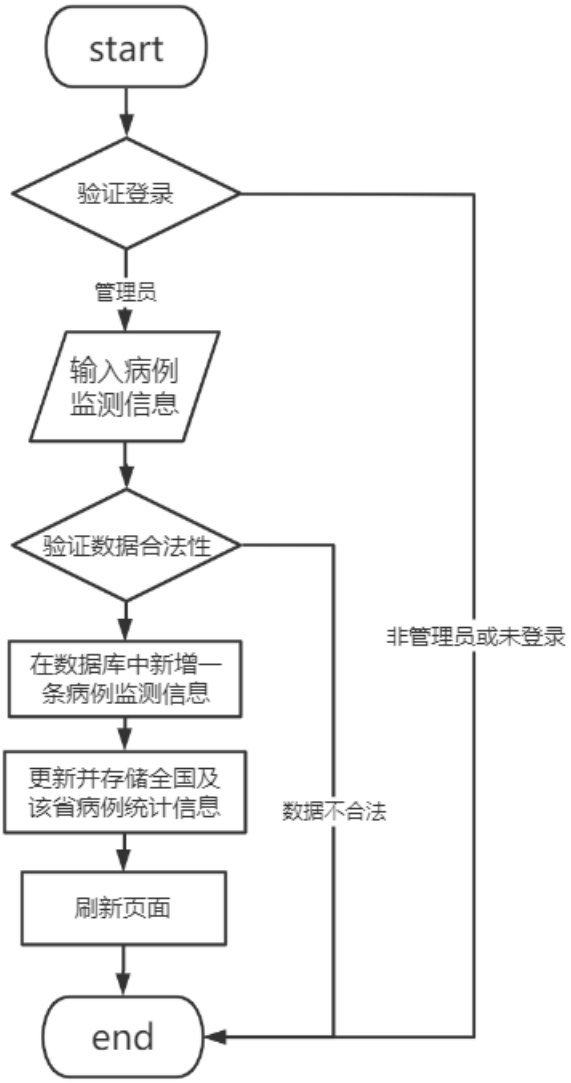
### 3.2.5 输出项

| 名称 | 标识 | 类型和格式 | 输出方式 |
|---|---|---|---|
| 更新结果 | EpidemicUpdateResult | Bool | 由脚本输出 |

### 3.2.6 设计方法（算法）

```
1.  if 管理员账号已登录
2.      连接数据库
3.      输入今日疫情监测信息
4.      检测输入信息是否合法（如信息是否完整、主键是否重复等）
5.      if 信息验证合法
6.          存储输入的更新信息
7.          更新该省和全国病例监测的统计情况
8.          返回成功信息（刷新界面）
9.      else
10.         返回失败信息（信息不合法）
11. else 返回失败信息（跳转到登录界面）
```

### 3.2.7 流程图




### 3.2.8 测试计划

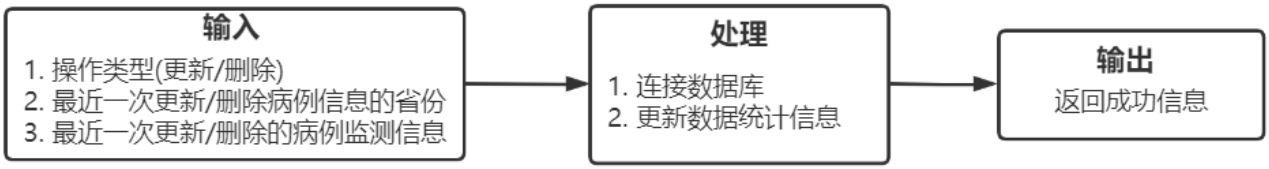
| 输入数据 | 预期结果 |
| --- | --- |
| 未登录，输入信息 | 返回错误信息，跳转到登录界面 |
| 输入信息不合法 | 返回错误信息，输入信息不合法 |
| 已登录，输入信息合法 | 返回成功信息，更新成功 |

## 3.3 数据统计

### 3.3.1 模块概述

这个模块用于计算全国及各省现存确诊总人数、现有疑似、境外输入总人数、现存无证感染总人数、累计确诊人数、累计死亡人数、累计治愈人数。每当有系统管理员发布新的病例监测信息或者删除已有的病例监测信息时，都会运行这个模块更新统计信息。

### 3.3.2 IPO 图




### 3.3.3 功能

统计全国各类病例总人数。

### 3.3.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 操作类型 | OperateType | Int | 外部输入 |
| 最近一次更新病例监测信息的省份 | Province | Int | 外部输入->脚本转换 |
| 最近一次更新的病例监测信息 | NewCase | Int | 外部输入 |

### 3.3.5 输出项

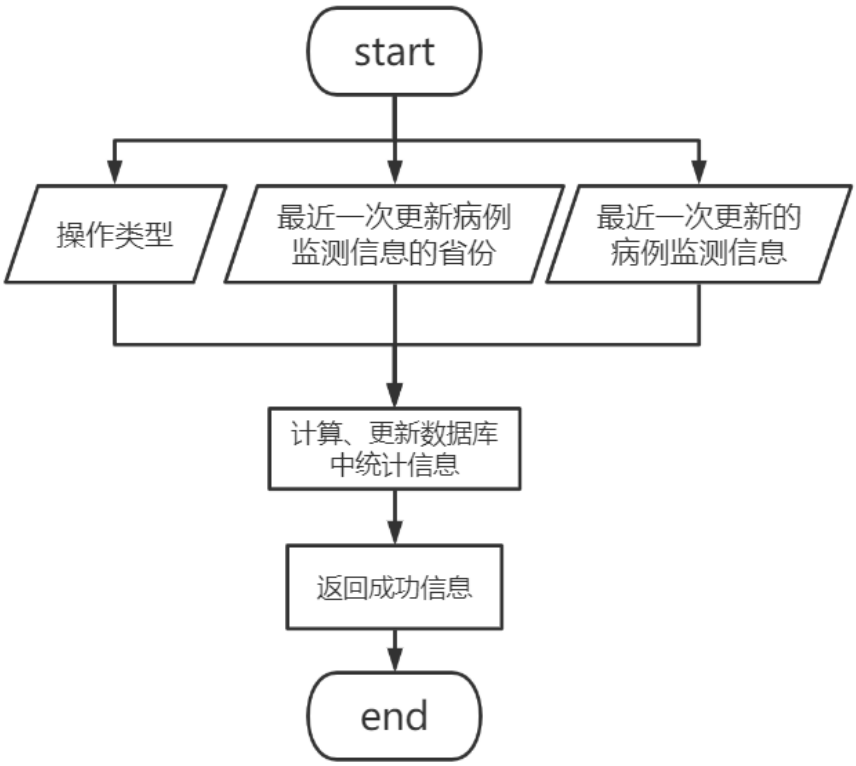
| 名称 | 标识 | 类型和格式 | 输出方式 |
| --- | --- | --- | --- |
| 统计结果 | CaseUpdateResult | Bool | 由脚本输出 |

### 3.3.6 设计方法（算法）

```
1.  创建数据库连接
2.  switch (操作类型):
3.      case (更新):
4.          //更新数据库中该省/全国各类病例统计信息
5.          该省/全国现存确诊人数 := 该省/全国现存确诊人数 + 今日确诊人数 - 今日死亡人数 - 今日治愈人
    数
6.          该省/全国现存疑似人数 := 该省/全国现存疑似人数 + 今日疑似人数变化
7.          该省/全国境外输入总人数 := 该省/全国境外输入总人数 + 今日境外输入人数
8.          该省/全国现存无证感染人数 := 该省/全国现存无证感染人数 + 今日现存无证感染人数变化
9.          该省/全国累计确诊人数 := 该省/全国累计确诊人数 + 今日确诊人数
10.         该省/全国累计死亡人数 := 该省/全国累计死亡人数 + 今日死亡人数
11.         该省/全国累计治愈人数 := 该省/全国累计治愈人数 + 今日治愈人数
12.         break
13.     case (删除):
14.         该省/全国现存确诊人数 := 该省/全国现存确诊人数 - 今日确诊人数 + 今日死亡人数 + 今日治愈人
    数
15.         该省/全国现存疑似人数 := 该省/全国现存疑似人数 - 今日疑似人数变化
16.         该省/全国境外输入总人数 := 该省/全国境外输入总人数 - 今日境外输入人数
17.         该省/全国现存无证感染人数 := 该省/全国现存无证感染人数 - 今日现存无证感染人数变化
18.         该省/全国累计确诊人数 := 该省/全国累计确诊人数 - 今日确诊人数
19.         该省/全国累计死亡人数 := 该省/全国累计死亡人数 - 今日死亡人数
```

```
20.         该省/全国累计治愈人数 := 该省/全国累计治愈人数 - 今日治愈人数
21.         break
22. 返回更新成功信息
```

### 3.3.7 流程图




### 3.3.8 测试计划

| 输入数据 | 预期结果 |
| --- | --- |
| 发布新的病例监测信息 | 返回成功信息，刷新页面 |
| 删除一条病例监测信息 | 返回成功信息，刷新页面 |

## 3.4 生成疫情地图

### 3.4.1 模块概述

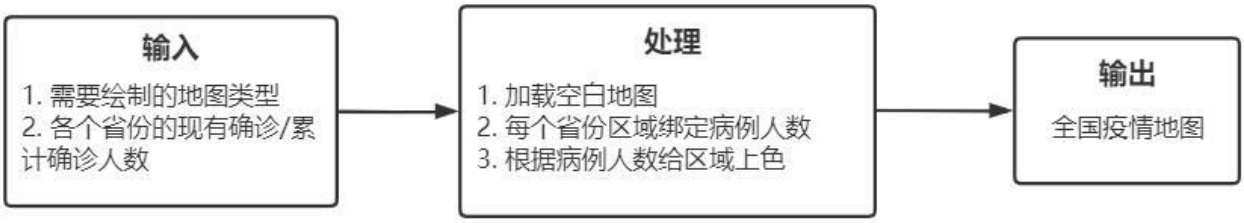
本模块的功能是生成全国疫情地图。根据各省现存确诊总人数（排除治愈、死亡）和累计确诊人数（包含治愈、死亡），地图上不同省份区域将显示不同颜色。病例数及其对应显示的颜色如下表所示。

表 1 疫情地图中病例数与地图颜色对应关系

| 病例数 / 例 | 颜色（十六进制） |
| --- | --- |
| ≥10000 | #C00000 |
| 1000-9999 | #F04848 |

| | |
|---|---|
| 100-999 | #F07860 |
| 10-99 | #FFA890 |
| 1-9 | #FFD8D8 |
| 0 | #FFFFFF |

### 3.4.2 IPO 图




### 3.4.3 功能

可视化展示全国各省疫情情况，包括现有确诊人数和累计确诊人数。

### 3.4.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
|---|---|---|---|
| 需要生成的疫情地图类型 | MapType | Int | 外部选择->脚本转换 |

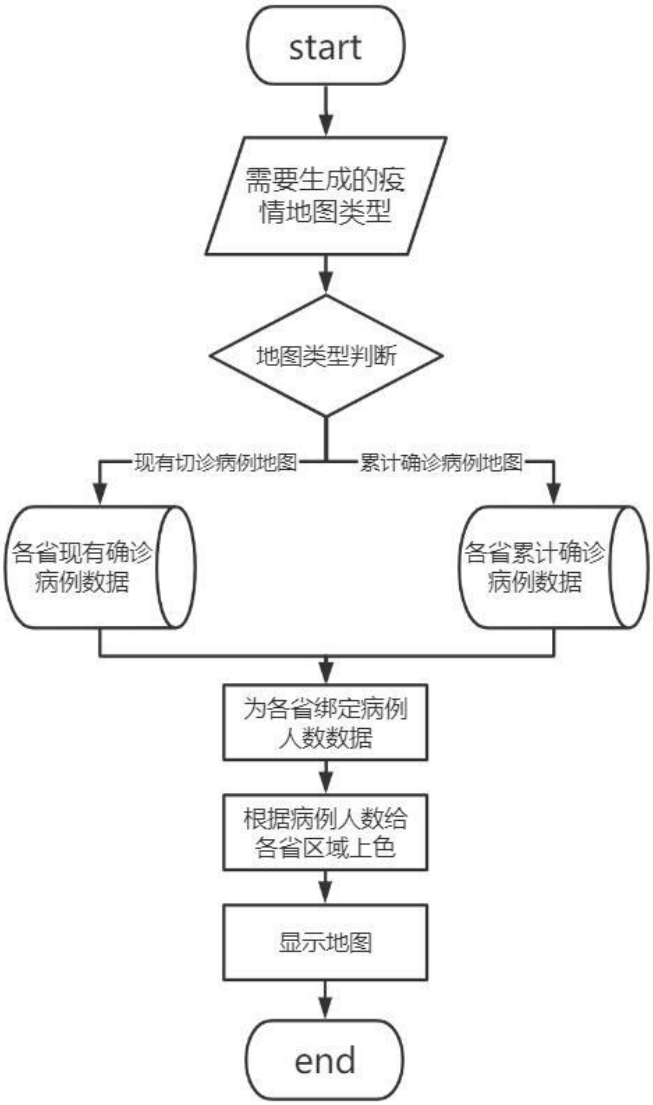
### 3.4.5 输出项

| 名称 | 标识 | 类型和格式 | 输出方式 |
|---|---|---|---|
| 疫情地图生成结果 | MapGenerationResult | Bool | 由脚本输出 |

### 3.4.6 设计方法（算法）

```
1.  if 生成全国现有确诊病例地图
2.      连接数据库
3.      读取各省现有确诊病例数据
4.      加载空白中国地图
5.      每个省份区域绑定其确诊病例人数
6.      根据病例人数，给对应省份区域设置不同颜色
7.      显示疫情地图
8.      返回地图生成成功信息（刷新现有确诊地图显示页面）
9.  else if 生成全国累计确诊病例地图
10.     连接数据库
11.     读取各省累计确诊病例数据
12.     加载空白中国地图
13.     根据病例人数，给对应省份区域设置不同颜色
14.     显示疫情地图
15.     返回地图生成成功信息（刷新累计确诊地图显示页面）
```

### 3.4.7 流程图




### 3.4.8 测试计划

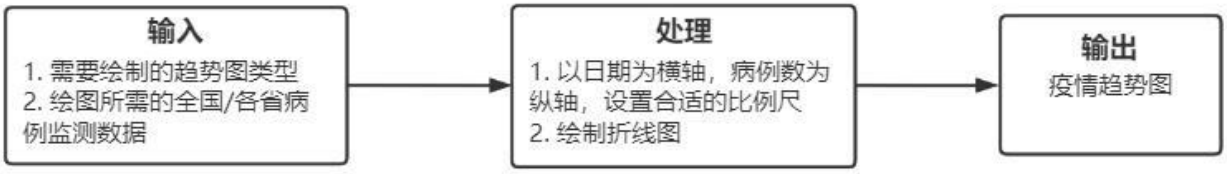
| 输入数据 | 预期结果 |
| --- | --- |
| 选择生成全国现有确诊病例地图 | 返回成功信息，展示全国现有确诊病例地图 |
| 选择生成全国累计确诊病例地图 | 返回成功信息，展示全国累计确诊病例地图 |

## 3.5 生成疫情趋势图

### 3.5.1 模块概述

本模块的功能是生成疫情变化趋势图，包括全国新增确诊/新增境外输入确诊趋势图、全国现有确诊/累计确诊趋势图、全国累计治愈/死亡趋势图，以及各省新增确诊/新增境外输入趋势图、各省现有确诊/累计确诊趋势图和各省累计治愈/死亡趋势图。

### 3.5.2 IPO 图




### 3.5.3 功能

绘制疫情趋势图，可视化展示全国疫情变化趋势。

### 3.5.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 需要生成的趋势图类型 | TrendChartType | Int | 外部选择->脚本转换 |
| 生成趋势图所需数据 | CaseData | Int | 由脚本输入 |

### 3.5.5 输出项

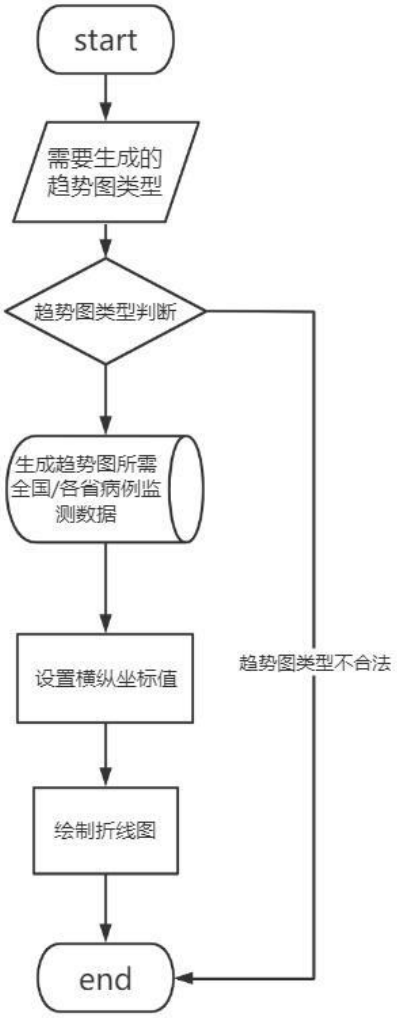
| 名称 | 标识 | 类型和格式 | 输出方式 |
| --- | --- | --- | --- |
| 趋势图生成结果 | ChartGenerationResult | Bool | 由脚本输出 |

### 3.5.6 设计方法（算法）

```
1.  连接数据库
2.  switch (需要生成的趋势图类型)
3.      case (全国新增确诊/新增境外输入确诊趋势图):
4.          从数据库读取数据（全国每日新增确诊人数、新增境外输入人数）
5.          break
6.      case (全国现有确诊/累计确诊趋势图):
7.          从数据库读取数据（全国每日现有确诊人数、累计确诊人数）
8.          break
9.      case (全国累计治愈/死亡趋势图):
10.         从数据库读取数据（全国每日累计治愈人数、累计死亡人数）
11.         break
12.     case (各省新增确诊/新增境外输入趋势图):
13.         确定目标省份
14.         从数据库读取数据（该省份每日新增确诊人数）
15.         break
16.     case (各省现有确诊/累计确诊趋势图):
17.         确定目标省份
18.         从数据库读取数据（该省份每日累计确诊人数）
19.         break
20.     case (各省累计治愈/死亡趋势图):
21.         确定目标省份
```

```
22.             从数据库读取数据（该省份每日累计治愈人数、死亡人数）
23.             break
24. 以日期为 x 轴，病例数为 y 轴，设置合适的坐标系
25. 根据数据绘制趋势图
26. 显示趋势图
27. 返回趋势图生成成功信息
```

### 3.5.7 流程图




### 3.5.8 测试计划

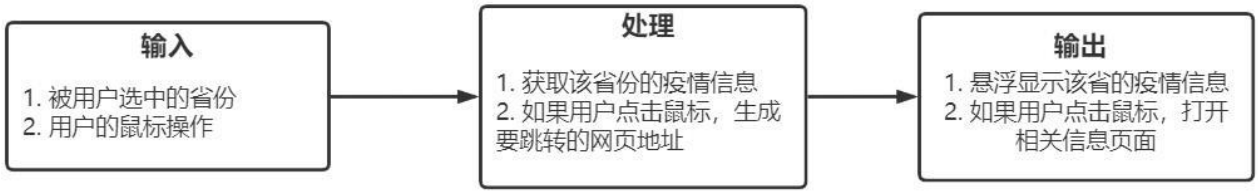
| 输入数据 | 预期结果 |
| --- | --- |
| 生成全国新增确诊/新增境外输入确诊趋势图 | 返回成功信息，展示全国新增确诊/新增境外输入确诊趋势图 |
| 生成全国现有确诊/累计确诊趋势图 | 返回成功信息，展示全国现有确诊/疑似/累计确诊趋势图 |
| 生成全国累计治愈/死亡趋势图 | 返回成功信息，展示全国累计治愈/死亡趋势图 |
| 生成某省新增确诊/新增境外输入趋势图 | 返回成功信息，展示某省新增趋势图 |
| 生成某省现有确诊/累计确诊趋势图 | 返回成功信息，展示某省累计确诊趋势图 |
| 生成某省累计治愈/死亡趋势图 | 返回成功信息，展示某省累计治愈/死亡趋势图 |

## 3.6 查看疫情地图

### 3.6.1 模块概述

该模块用于向用户展示疫情地图的详细数据。当用户在查看疫情地图时，该模块始终监测用户的鼠标操作。当用户将鼠标移动到疫情地图内的某个省份区域内时，该省份区域将被高亮显示，同时悬浮显示该省的名字、现有确诊人数或者累计确诊人数。当鼠标点击某个省份区域时，将跳转到该省的疫情信息详细页面。

### 3.6.2 IPO 图




### 3.6.3 功能

用户可以查看疫情地图及相关信息。

### 3.6.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
|---|---|---|---|
| 被选中的省份 | SelectedProvince | Int | 外部选择->脚本转换 |
| 鼠标动作事件 | MouseEvent | Int | 外部输入->脚本转换 |

### 3.6.5 输出项

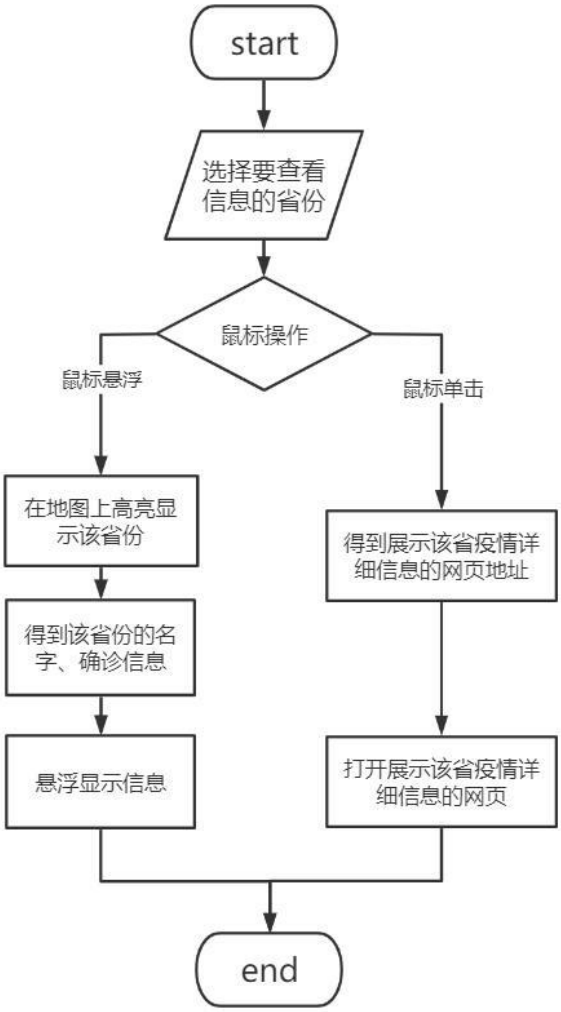
| 名称 | 标识 | 类型和格式 | 输出方式 |
|---|---|---|---|
| 操作状态 | MouseEventResult | Bool | 脚本输出 |

### 3.6.6 设计方法（算法）

```
1.  获取鼠标位置
2.  if 鼠标悬停在某省区域内
3.      高亮显示该省区域
4.      获取该省病例信息
5.      悬浮显示该省名字、病例信息
6.      返回鼠标响应事件结果(响应成功/响应失败)
7.
8.  if 鼠标点击某省区域
9.      得到展示该省疫情详细信息的网页
```

10. 打开展示该省疫情详细信息的网页
11. 返回鼠标响应事件结果(响应成功/响应失败)

### 3.6.7 流程图




### 3.6.8 测试计划

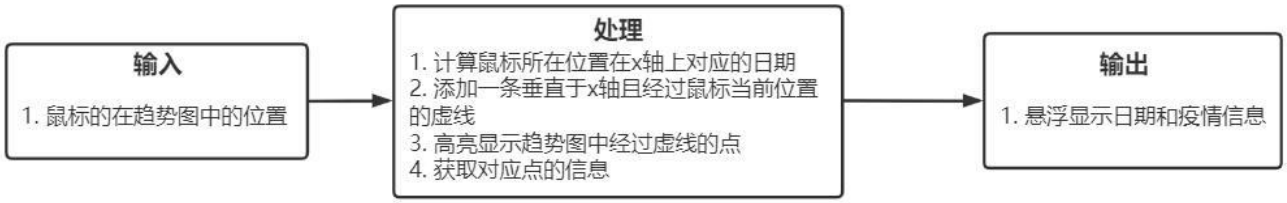
| 输入数据 | 预期结果 |
|---|---|
| 鼠标悬停在地图上 A 省区域内 | 返回成功信息，在 A 省区域上有悬浮面板展示 A 省疫情信息 |
| 鼠标单击地图上 A 省区域 | 返回成功信息，打开展示 A 省疫情详细信息的网页 |

## 3.7 查看疫情趋势图

### 3.7.1 模块概述

该模块用于向用户展示疫情趋势图的详细数据。当用户在查看疫情趋势图时，该模块始终监测用户的鼠标操作。当用户将鼠标移动到疫情趋势图上任意位置，将生成一条经过该位置的虚线。位于该虚线的趋势线上的点将被高亮显示，同时悬浮显示该位置对应的日期、以及被高亮标出的点对应的信息。

### 3.7.2 IPO 图




### 3.7.3 功能

用户可以全国或各省查看疫情趋势图及相关信息。

### 3.7.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 鼠标水平位置 | MousePosition | Double | 外部输入->脚本转换 |

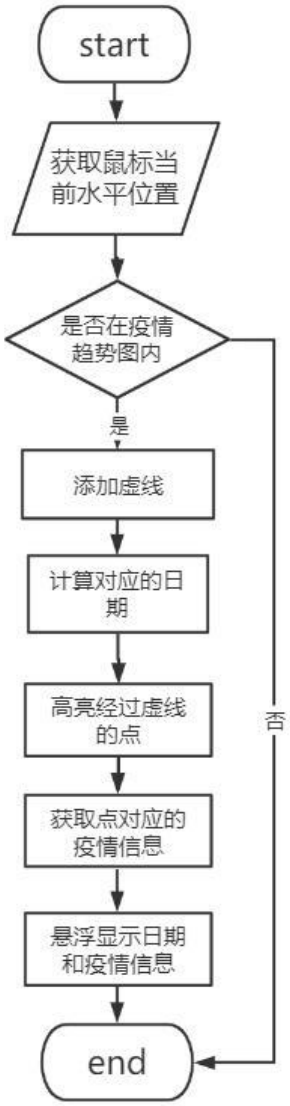
### 3.7.5 输出项

| 名称 | 标识 | 类型和格式 | 输出方式 |
| --- | --- | --- | --- |
| 操作状态 | ShowChartResult | Bool | 脚本输出 |

### 3.7.6 设计方法（算法）

```
1.  获取鼠标当前水平位置
2.  if 鼠标位于疫情趋势图内
3.      添加一条经过鼠标当前位置的垂直于 x 轴的虚线
4.      高亮经过该虚线的点
5.      计算该位置在 x 轴上对应的日期
6.      获取所有趋势线上该日期对应的疫情信息
7.      悬浮显示日期和疫情信息
8.      返回操作成功结果
9.  else   返回操作失败结果
```

### 3.7.7 流程图




### 3.7.8 测试计划

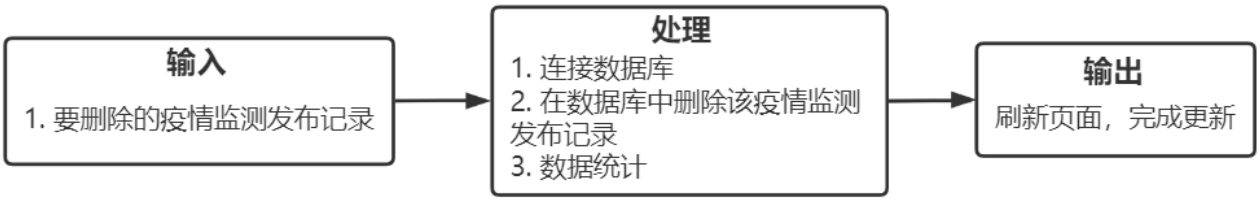
| 输入数据 | 预期结果 |
| --- | --- |
| 鼠标位于疫情趋势图外 | 返回失败结果，趋势图没有变化 |
| 鼠标位于疫情趋势图内 | 返回成功结构，悬浮显示对应日期的疫情信息 |

## 3.8 数据删除

### 3.8.1 模块概述

若某一天发布的数据有错误，系统管理员可以在该模块中删除疫情发布记录。删除记录后，疫情监测信息会重新进行数据统计。

### 3.8.2 IPO 图




### 3.8.3 功能

删除疫情监测信息发布记录。

### 3.8.4 输入项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 发布记录编号 | RecordID | Int | 外部选择->脚本转换 |

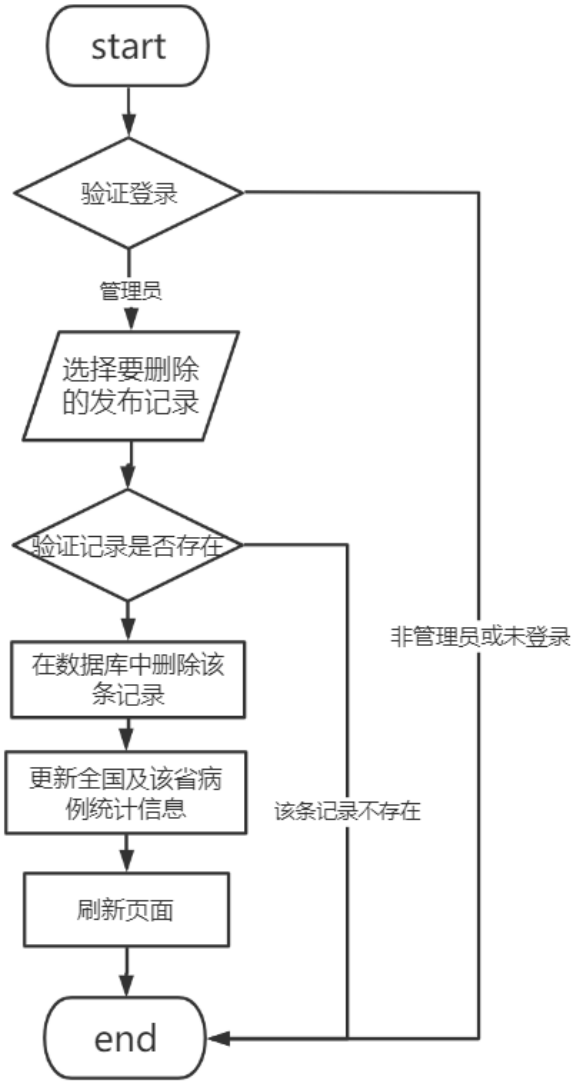
### 3.8.5 输出项

| 名称 | 标识 | 类型和格式 | 输入方式 |
| --- | --- | --- | --- |
| 操作结果 | DeleteResult | Int | 脚本输出 |

### 3.8.6 设计方法（算法）

```
if 管理员账号已登录
    连接数据库
    选择要删除的疫情监测信息发布记录
    if 该条记录存在
        删除记录
        更新该省和全国病例监测的统计情况
        返回成功信息（刷新界面）
    else
        返回失败信息（记录不存在）
else 返回失败信息（跳转到登录界面）
```

### 3.8.7 流程图

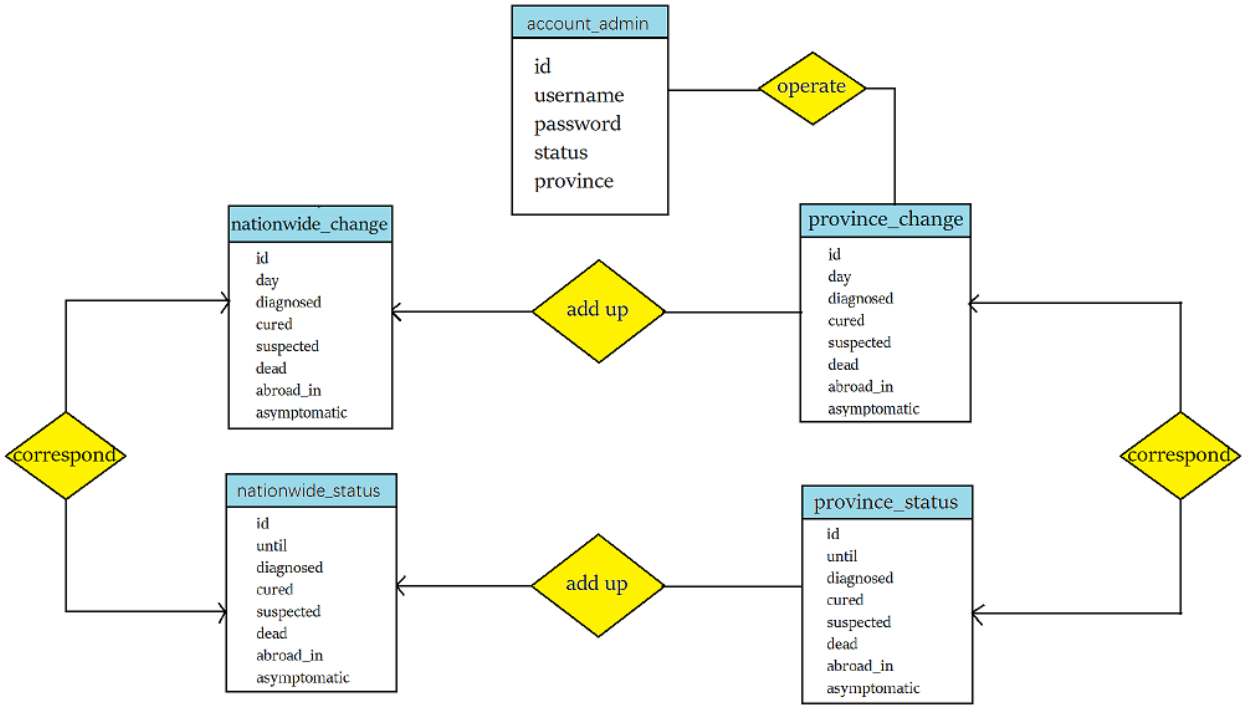



### 3.8.8 测试计划

| 输入数据 | 预期结果 |
| --- | --- |
| 未登录，输入信息 | 返回错误信息，跳转到登录界面 |
| 选择删除的记录不存在 | 返回错误信息，记录不存在 |
| 已登录，选择删除的记录存在 | 返回成功信息，删除成功 |

# 4 数据库设计

## 4.1 ER-图




## 4.2 逻辑结构设计

- 某省份疫情情况（累计）

epidemic_status_XXX (id, until, diagnosed, dead, suspected, cured, asymptomatic)

- 某省份疫情情况(每日新增)

epidemic_change_XXX (id, until, diagnosed, dead, suspected, cured, asymptomatic)

- 全国疫情情况（累计）

epidemic_status_nationwide (id, until, diagnosed, dead, suspected, cured, asymptomatic)

- 全国疫情情况（每日新增）

epidemic_change_nationwide (id, until, diagnosed, dead, suspected, cured, asymptomatic)

- 管理员账户

account_admin (id, username, password, status, province)

## 4.3 物理结构设计

### 4.3.1 省份疫情统计情况

| epidemic_status_XXX(province name) | | | | | |
|---|---|---|---|---|---|
| 字段 | 类型 | 能否为空 | 是否为主键 | 是否为外键 | 备注 |
| id | varchar(20) | × | √ | × | 记录唯一标识 id |
| until | date | × | × | × | 统计截至日期 |
| diagnosed | integer | × | × | × | 确诊病例数量 |
| dead | integer | × | × | × | 死亡病例数量 |
| suspected | integer | × | × | × | 疑似病例数量 |
| asymptomatic | integer | × | × | × | 无症状感染者数量 |
| cured | integer | × | × | × | 治愈病例数量 |
| abroad_in | integer | × | × | × | 境外输入病例数量 |

### 4.3.2 各省疫情情况每日变化

| epidemic_change_XXX(province) | | | | | |
|---|---|---|---|---|---|
| 字段 | 类型 | 能否为空 | 是否为主键 | 是否为外键 | 备注 |
| id | varchar(20) | × | √ | × | 记录唯一标识 id |
| day | date | × | × | × | 统计日期 |
| diagnosed | integer | × | × | × | 确诊病例数量 |
| dead | integer | × | × | × | 死亡病例数量 |
| suspected | integer | × | × | × | 疑似病例数量 |
| asymptomatic | integer | × | × | × | 无症状感染者数量 |
| cured | integer | × | × | × | 治愈病例数量 |
| abroad_in | integer | × | × | × | 境外输入病例数量 |

### 4.3.3 全国疫情统计情况

| epidemic_status_nationwide | | | | | |
|---|---|---|---|---|---|
| 字段 | 类型 | 能否为空 | 是否为主键 | 是否为外键 | 备注 |
| id | varchar(20) | × | √ | × | 记录唯一标识 id |
| until | date | × | × | × | 统计截至日期 |
| diagnosed | integer | × | × | × | 确诊病例数量 |
| dead | integer | × | × | × | 死亡病例数量 |
| asymptomatic | integer | × | × | × | 无症状感染者数量 |
| suspected | integer | × | × | × | 疑似病例数量 |
| cured | integer | × | × | × | 治愈病例数量 |
| abroad_in | integer | × | × | × | 境外输入数量 |

### 4.3.4 全国疫情每日变化：

| epidemic_change_nationwide | | | | | |
|---|---|---|---|---|---|
| 字段 | 类型 | 能否为空 | 是否为主键 | 是否为外键 | 备注 |
| id | varchar(20) | × | √ | × | 记录唯一表示 id |
| day | date | × | × | × | 统计截至日期 |
| diagnosed | integer | × | × | × | 确诊病例数量 |
| dead | integer | × | × | × | 死亡病例数量 |
| suspected | integer | × | × | × | 疑似病例数量 |
| asymptomatic | integer | × | × | × | 无症状感染者数量 |
| cured | integer | × | × | × | 治愈病例数量 |
| abroad_in | integer | × | × | × | 境外输入病例数量 |

### 4.3.5 管理员账户

| account_admin | | | | | |
|---|---|---|---|---|---|
| 字段 | 类型 | 能否为空 | 是否为主键 | 是否为外键 | 备注 |
| id | varchar(20) | × | √ | × | 唯一标识 |
| username | varchar(16) | × | √ | × | 唯一用户名 |
| password | varchar(16) | × | × | × | 密码 |
| status | integer | × | × | × | 账户状态，是否锁定 |
| province | varchar(100) | × | × | × | 所负责省份 |

# 5 接口设计

## 5.1 疫情信息首页界面设计

疫情地图是动态的，可放大，可缩小，鼠标移到具体省份可以显示省份疫情数据。管理员可通过右上角“后台管理”登录后台进行管理疫情发布信息。


## 5.2 登录界面设计

忘记密码可以找回，“记住我”功能只会记住用户名。


## 5.3 后台界面设计

点击“发布数据”，输入相应数据即可发布。


## 5.4 外部接口

通过设置 Diango 框架里的 settings.py 直接连接数据库。

## 5.5 内部接口

| 操作 | 查看省份病例数据 | 查看全国病例数据 | 发布数据 |
|---|---|---|---|
| 实现 | 登录数据库，查表 | 登录数据库，查表 | 管理员登录 |
| | 计算 | 计算 | 检查管理员表 |
| | 发送到页面 | 发送到页面 | 修改省疫情信息表 |

# 6 运行设计

## 6.1 运行模块组合

本子系统按照功能划分模块，每个模块又按流程划分为客户端界面，客户端脚本，服务器后台程序。功能模块之间会共享部分界面（导航栏等）。

## 6.2 运行控制

普通用户通过用户登录模块(将在后续调试中使用用户管理子系统小组的模块)，选择用户类型(用户登录、游客登录)。用户登录需要在表单中输入正确的账户密码，从而登录系统；游客登陆则无需输入信息，直接跳转为游客身份。本子系统主要用于病例检测结果的呈现，故普通用户和游客拥有的模块功能无差别。

管理员通过本子系统疫情信息首页右上角的“后台管理”，输入管理员账号密码进行登录。管理员拥有发布相应省份疫情数据信息的权限。

- **若登录者为管理员**

**数据发布**

管理员登录状态下，可以进入“数据发布模块”。表单中，省份一栏默认为该管理员负责的省份。可以选择提交模式和修改模式。提交模式向数据库中插入一条新的记录，修改模式中的修改操作覆盖数据库中原有的记录，删除操作则删去一条记录。

选择提交模式，在表单中填写该省新增确诊、疑似、境外输入、无证感染、治愈、死亡人数后可点击提交，否则提示某项不能为空。

若当日或过去数据填写错误，可选择删除或者修改。输入正确的数据后提交修改，同样限制新增确诊、疑似、境外输入、无症状感染、治愈、死亡人数不可为空。

**数据统计**

在每次进行提交/修改/删除操作后，数据统计模块通过数据库的触发器计算统计数据，并在前端

实时显示最新的全国及该省现存确诊患者、疑似患者、无症状感染者、境外输入病例数量，以及累计确诊、死亡、治愈人数等统计数据。

- **若登录者为普通用户/游客**

**查看疫情地图**

在病例监测子系统的主界面，用户可以查看全国疫情统计数据、全国疫情地图。疫情地图可以进行放大缩小。鼠标悬停于地图中某省份上，悬浮显示对应省份今日新增等数据信息。鼠标单击地图中某省份，跳转至该省疫情信息详情页面。

**查看疫情趋势图**

在病例监测子系统的主界面，用户可以查看全国新增确诊/新增境外输入确诊趋势图、全国现有确诊/累计确诊趋势图、全国累计治愈/死亡趋势图等图，在各省详细疫情情况页面，用户可以看到各省新增确诊/新增境外输入趋势图、各省现有确诊/累计确诊趋势图和各省累计治愈/死亡趋势图。

用户将鼠标移动到疫情趋势图上的任意位置，将生成一条经过该位置的虚线。位于该虚线的趋势线上的点将被标出，同时悬浮显示该位置对应的日期以及病例数等信息。

# 7 系统出错设计

## 7.1 出错信息

| 系统输出信息的形式 | 含义 | 处理办法或相应对策 |
|---|---|---|
| **数据库连接失败** | ① 数据库配置出错<br>② 数据库连接数超过上限 | ① 修改数据库配置<br>② 限制并非访问量 |
| **用户名密码错误** | ① 用户忘记密码<br>② 被他人恶意篡改密码 | 返回相应的错误信息给用户，通过登录界面的“找回密码”重拾 |
| **服务器暂时无法访问** | ① 服务器正在维护中<br>② 短时间内有大量流量导致服务器瘫痪 | 对于②的情况，联系系统管理员进行紧急处理 |
| **非法指令** | 部分用户企图访问管理员界面或后台程序，窃取网站 | 限制普通用户越权访问，通过各种手段保护后台数据 |
| **磁盘损坏** | 磁盘受损 | 对磁盘与数据库进行周期性的备份 |
| **并发访问量过大** | 多并发访问，用户流量大 | 使用负载均衡技术 |
| **网站页面加载慢** | 加载机制有误 | HTML 静态化 |

## 7.2 补救措施

### 7.2.1 系统恢复

系统崩溃后，根据系统运行日志恢复系统和数据，并重新启动系统。

### 7.2.2 定时备份

（1）周期性地备份数据库中的数据，将其存储在更稳定的介质上，并及时进行校对、更新。

（2）将工程代码通过 GitHub 进行完善的版本管理。

### 7.2.3 人工操作

当出现紧急情况时，数据库管理员人工地对数据库中数据进行修改，并做相应的记录。

## 7.3 系统维护设计

由于实验条件有限，我们并不能提供专门的服务器运行系统，故将利用配置较高的 PC 作为服务器，保证服务器以及客户端间网络畅通即可。

（1）用户在该系统执行操作时应该留下痕迹，以方便检查系统是否被恶意篡改。同时系统管理员定时查看系统日志，统计非法攻击来源和次数，并针对相应攻击加强安全防范措施。

（2）系统管理员的登录不仅需要账户密码，还需要识别 IP，禁止非白名单 IP 的任何访问。

（3）系统维护人员及时更新技术漏洞，通过各种手段防止各种对系统的攻击，增强代码的可靠性。

（4）定期维护数据库，涉及到检查数据库表、检查日志文件等，确保数据库内数据的正确性。

(5)在可能出错的地方使用 try-catch 语句捕获异常,并输出相应的出错信息和可能的处理方法提示。

# 8 需求回溯

## 8.1 功能性需求回溯

| 需求ID | 需求简述 | 对应的设计模块 | 实现方式 |
| --- | --- | --- | --- |
| A01 | 用户身份有效性核验 | 用户管理模块 | 页面加载时，服务器验证 Session 是否过期，如果过期则重定向到所有子系统统一的登录界面。否则用户将根据 Session 中存储的身份信息，拥有不同的模块功能。 |
| A02 | 为管理员提供发布疫情数据的操作 | 数据发布模块 | ①服务端验证Session后确认用户身份为管理员时，自动以数据库管理员身份连接数据库。<br>②管理员在前端执行提交操作时，检查表单是否全部非空。若否，则弹出对话框提示“xx不能为空！”；若是，则调用sql语句插入一条新记录。 |

| | | | |
|---|---|---|---|
| | | | ③管理员在前端执行修改操作时，检查表单是否全部非空。若否，则弹出对话框提示“xx不能为空！”；若是，调用sql语句删除数据库中原有的记录（若存在）并重新添加。<br>④管理员在前端执行删除操作时，调用sql语句删除数据库中原有的记录（若存在）。 |
| A03 | 为管理员提供统计疫情数据的功能 | 数据统计模块 | 当管理员执行提交或修改操作后，触发器启动，自动计算疫情数据的各项统计结果，存储至数据库中的统计结果表，并在前端显示统计数据结果。 |
| A04 | 为普通用户或游客提供可以查看疫情地图的界面 | 疫情地图模块 | ①用户可以在主界面查看全国疫情统计数据、全国疫情地图。<br>②鼠标悬停于地图上，悬浮显示对应省份今日新增等数据信息。<br>③鼠标单击地图中某省份，打开该省疫情信息详情页面。 |
| A05 | 为普通用户或游客提供可以查看疫情趋势图的界面 | 疫情趋势图模块 | ①在全国疫情趋势图页面，用户可以选择查看全国新增确诊/新增境外输入确诊趋势图、全国现有确诊/累计确诊趋势图、全国累计治愈/死亡趋势图。<br>②在各省详细疫情情况页面，用户可以看到各省新增确诊/新增境外输入趋势图、各省现有确诊/累计确诊趋势图和各省累计治愈/死亡趋势图。<br>③用户将鼠标移动到疫情趋势图上的任意位置，将生成一条经过该位置的虚线。位于该虚线的趋势线上的点将被标出，同时悬浮显示该位置对应的日期以及病例数等信息。 |

## 8.2 性能及安全需求

| 需求ID | 需求简述 | 实现方式 |
|---|---|---|
| B01 | 【安全性】用户账户安全 | 使用 OAuth2.0 标准协议 |
| B02 | 【安全性】数据库数据保护，防止用户恶意访问数据库或者破坏数据库 | ①使用占位符进行查询参数绑定<br>②数据库根据不同的用户赋予不同的最小需要的操作权限。 |
| B03 | 【性能】能够应对数据库访问量过大的情况以及通过大量访问量来实现攻击的手段 | 对于频繁访问数据库的操作，需要建立索引，使用 redis 缓存来进行优化。 |
| B04 | 【性能】客户端通过网页展现给用户一个友好、易于操作的界面 | 通过 React 和定义外部 CSS 样式表实现扁平化样式与栅格排版，使用 JavaScript 实现数据的动态显示和交互。 |