COVID-19 数据发布、订阅、分析预测系统

小组成员：2051840 梁厚，2051849 王崧宇，2051374 吴雨阳，2052134 刘治华，1953729 吴浩泽

目录

1. 项目简介

1.1 项目背景

COVID-19（新型冠状病毒）是一种新型的严重呼吸系统疾病。从 2019 年 12 月和 2020 年 1 月开始，在中国武汉开始大范围传播，对全国人民的人身安全和社会经济造成了严重的威胁和损失。

截至 2022 年年末，全球疫情的增速和确诊人数仍在上升，疫情本身仍未完全得到控制。除去医疗卫生等直接防疫手段，疫情防控同样需要基于疫情的实时数据不断升级改善防疫政策。当前，各国疾控中心均需要基于大数据的疫情数据统计系统，用于数据统计，数据分析以及数据预测等需求。

1.2 项目目标

本项目旨在为医务工作者和普通民众构建出一个疫情大数据可视化系统，帮助疫情防控机构更直观的展示中国各省份的详细疫情数据，同时利用现有算法和技术对未来疫情趋势进行预测以帮助人们更好的制定疫情防控政策，提前预知可能风险。

本项目以物联网为出发点，同样希望通过疫情数据的发布，代理和处理来模拟一个物联网的场景，并尝试将 MQTT 通信协议应用到项目当中。MQTT 本身的低带宽和即时性，同样和本项目的基本需求相适配，利用现有的 MQTT 平台将此协议应用到本项目中能够帮助我们更好地去理解物联网。

2. 功能介绍

2.1 展示当前时间

1. 用例描述

用例名称	展示当前时间
标识符	UC01
说明	页面右上角可以展示当前时间
前置条件	用户已成功进入疫情可视化系统
后置条件	无

基本操作流程	1.系统请求当前时间 2.系统将时间渲染到右上角位置
可选操作流程	无

1. 功能截图

2.2 展示国内省级区域疫情数据

1. 用例描述

用例名称	展示国内省级区域疫情数据
标识符	UC02
说明	页面借助中国地图的形式展示全国各个省份新增确诊人数
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	1.用户将鼠标移动至地图上对应的省份区域 2.系统向后端请求相应省份对应的新增确诊人数并将数据渲染至地图相应位置，并按照不同的人数分级，赋予不同区域不同饱和度的颜色
可选操作流程	无

1. 功能截图

2.3 展示国内市级区域疫情数据

1. 用例描述

用例名称	展示国内市级区域疫情数据
标识符	UC03
说明	页面借助地图的形式展示选定省份各市新增确诊人数
前置条件	用户已成功进入疫情可视化系统并在中国地图上，单击某一省份，下钻进入省级地图
后置条件	无
基本操作流程	1.用户将鼠标移动至地图上对应的市级区域 2.系统向后端请求相应区域对应的新增确诊人数并将数据渲染至地图相应位置，并按照不同的人数分级，赋予不同区域不同饱和度的

	颜色
可选操作流程	无

1. 功能截图

2.4 国内疫情实时新闻播报

1. 用例描述

用例名称	国内疫情实时新闻播报
标识符	UC04
说明	页面通过在左侧悬浮轮播展示最新的疫情新闻播报
前置条件	用户已成功进入疫情可视化系统

后置条件	无
基本操作流程	1.系统向接口请求最新相关新闻和对应时间 2.系统将新闻渲染到左侧悬浮框上
可选操作流程	用户点击新闻跳转到新闻网站

1. 功能截图

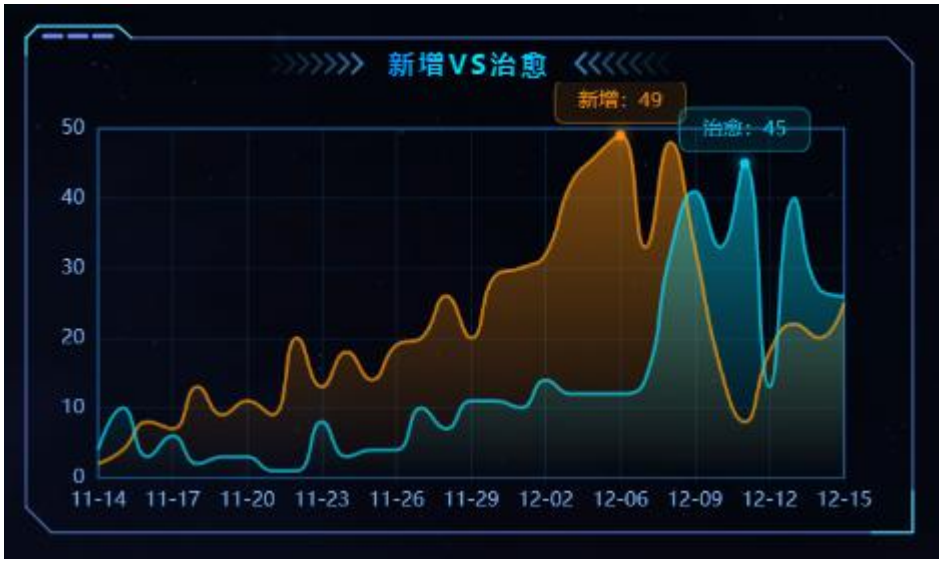
2.5 国内疫情随时间变化展示

1. 用例描述

用例名称	国内疫情随时间变化展示

标识符	UC05
说明	页面通过曲线对比图展示当前所选省份的新增人数、治愈人数曲线图
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	1.系统向接口请求所选省份的历史疫情数据 2.系统将得到的数据渲染到曲线对比图中
可选操作流程	1.默认展示上海市的新增人数曲线和治愈人数曲线 2.当用户选择其他省份进行下钻时，曲线图会随之切换

1. 功能截图



2.6 国内疫情趋势预测

1. 用例描述

用例名称	国内疫情趋势预测
标识符	UC06
说明	页面将对于国内疫情新增确诊和新增治愈人数的预测数据渲染

	到曲线对比图中
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	1.系统向接口请求预测数据 2.系统将预测数据渲染到曲线对比图中
可选操作流程	用户选择不同的省份时，曲线图需要随着切换

1. 功能截图

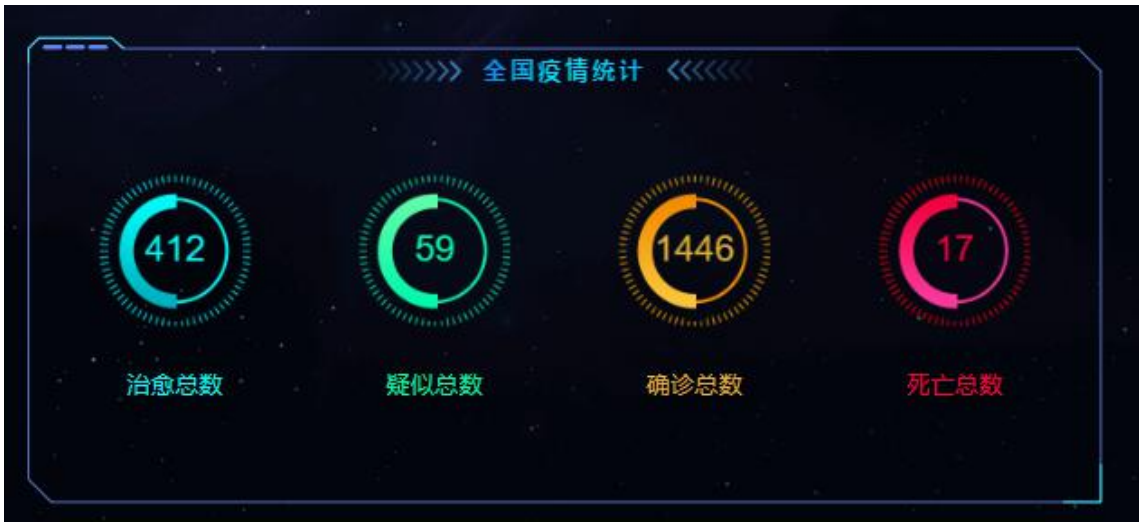
2.7 国内疫情数据统计

1. 用例描述

用例名称	国内疫情数据统计
标识符	UC07
说明	页面展示全国治愈总人数、疑似确诊总数、确诊总人数、死亡总人数
前置条件	用户已成功进入疫情可视化系统
后置条件	无

基本操作流程	1.系统向接口请求全国治愈总人数、疑似确诊总数、确诊总人数、死亡总人数数据 2.系统将数据渲染到界面中
可选操作流程	无

1. 功能截图



2.8 国内疫情数据比例分布展示

1. 用例描述

用例名称	国内疫情数据比例分布展示
标识符	UC08
说明	页面展示全国治愈总人数、疑似确诊总数、确诊总人数、死亡总人数这四者之间的比例分布
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	1.系统向接口请求全国治愈总人数、疑似确诊总数、确诊总人数、死亡总人数数据并计算各项数据所占比例 2.系统将比例计算结果渲染到界面中

可选操作流程	无

1. 功能截图

2.9 国内疫情确诊人数地区排名

1. 用例描述

用例名称	国内疫情确诊人数地区排名
标识符	UC09
说明	页面展示全国疫情新增确诊人数最多的 8 个市级区域排名
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	1.系统向接口请求全国疫情新增确诊人数最多的 8 个市级区域和对应的新增人数 2.系统将数据渲染到排名表中
可选操作流程	无

1. 功能截图

2.10 对应省份确诊/死亡/治愈人数

1. 用例描述

用例名称	对应省份的确诊、死亡、治愈人数的总数和每天的数量
标识符	UC10
说明	通过上拉菜单和切换按钮选择查看对应的数据
前置条件	用户已成功进入疫情可视化系统
后置条件	无
基本操作流程	选择对应的省份请求数据，然后切换按钮选择想要的部分数据显示
可选操作流程	无

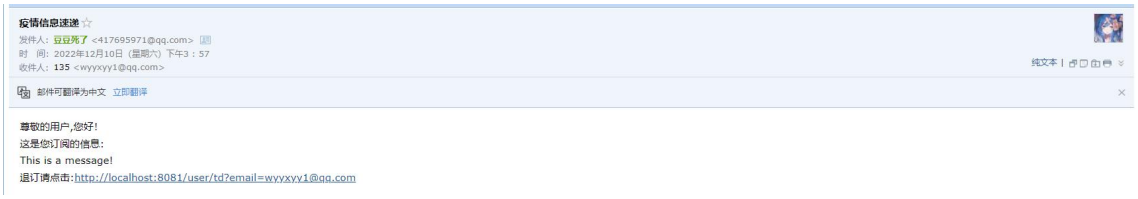
1. 功能截图

2.11 邮箱定时发送疫情信息

1. 用例描述

用例名称	邮箱定时发送疫情信息
标识符	UC11
说明	每天中午 12 点会往订阅人的邮箱发送疫情信息
前置条件	用户已订阅疫情消息
后置条件	无
基本操作流程	无
可选操作流程	无

1. 功能截图

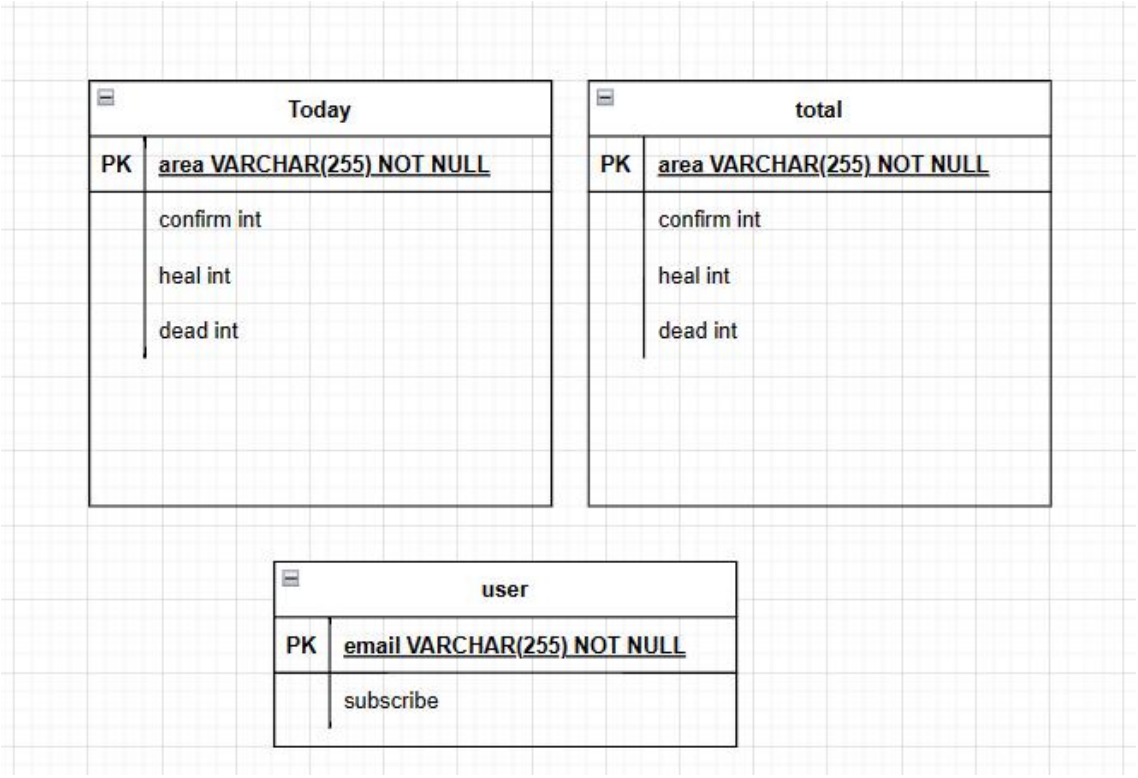


3. 数据库设计

3.1 数据库环境

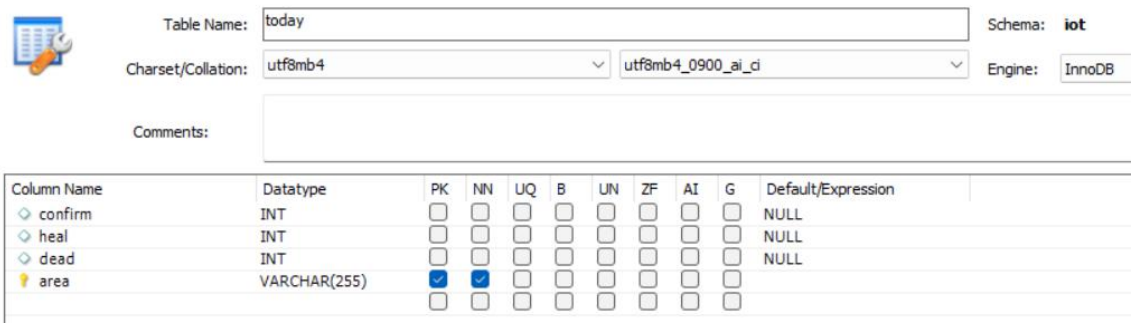
本次项目我们根据我们的需求，选择了较为轻量级的 Mysql 数据库。

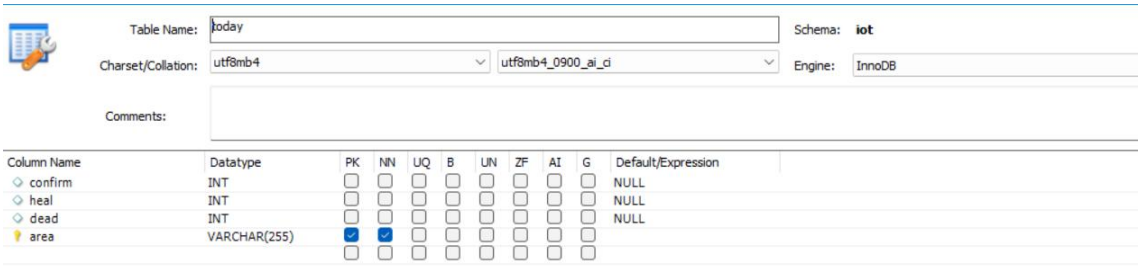
3.2 数据库框架

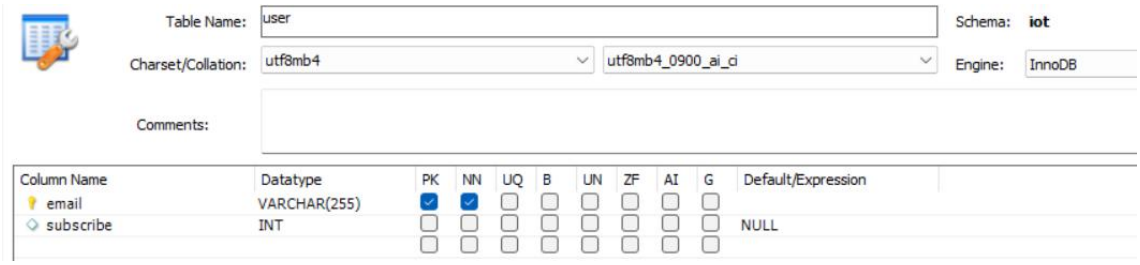


3.3 数据库详细信息

数据库具体表

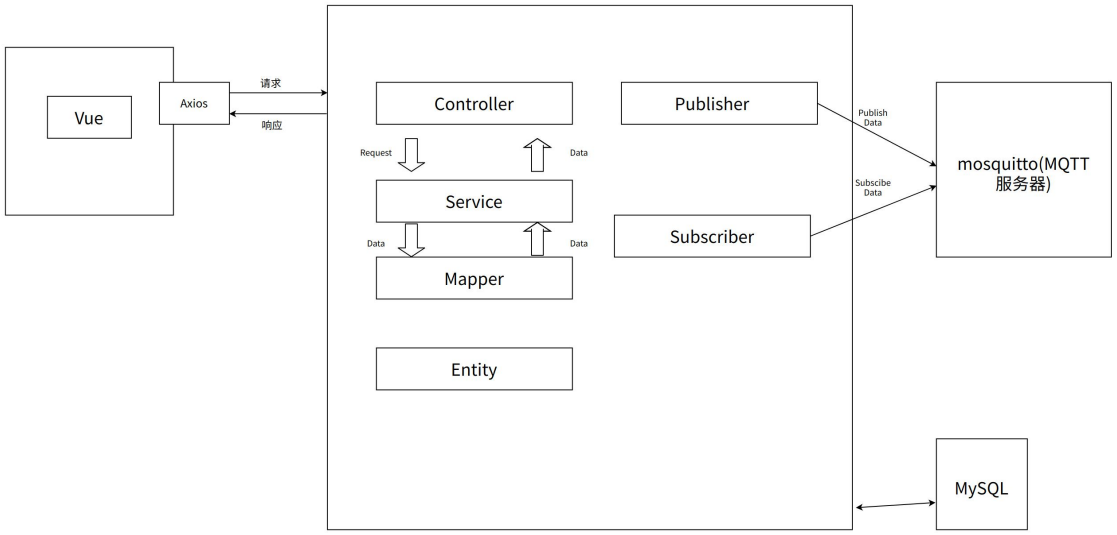






4. 框架设计

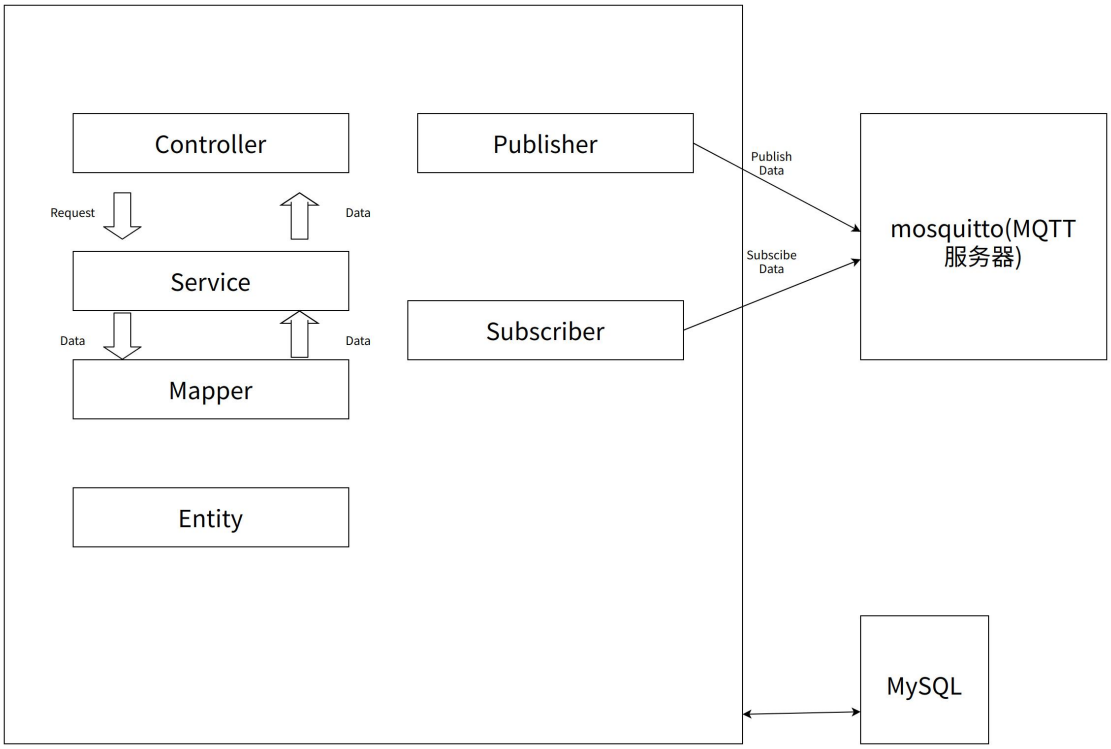
4.1 全局框架



4.2 前端框架

前端采用 Vue 框架，并通过 ElementUI，enchart 等组件库快速动态的大屏数据展示页面，通过 axios 发送 http 请求与后端项目互传数据。

4.3 后端框架



4.3.1 主要组件

后端主要采用了 Springboot 的框架搭建，它主要由以下的组件所构成：

- Controller：与外部世界交互并直接调用 Servicer 来处理业务逻辑；
- Servicer：处理业务逻辑。我们使用接口和 impl 来减少耦合；
- Mapper(DAO) :我们主要利用 jpa 框架封装好的 repository 的接口类进行对数据库的查询；
- Entity：根据业务逻辑和数据库进行设计，主要用于承载具体的数据。

4.3.2 MQTT 协议相关组件

此外我们还将 Springboot 框架和 MQTT 协议进行了整合，在 Springboot 框架中加入了 MQTT 协议的发布者和订阅者的类：

- publisher：作为 MQTT 协议的发布者，从网站上定时爬取各省份感染人数的数据，将消息发送至服务器；
- subscriber：作为 MQTT 协议的订阅者。从 MQTT 服务器上订阅消息，将数据保存到本地文件中。

4.3.3 其他实体类

除了必要的控件外，我们还设计了一些重要的实体类：

- common：存放作为返回数据的 json 类等

5. 项目算法和传输协议

5.1 核心算法

我们处理及预测数据采用的是最小二乘法来进行多项式曲线拟合，即：

假设给定的数据点和其对应的函数值为 $(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)$，需要做的就是得到一个多项式函数$f(x)=a_0x+a_1x^2+\cdots+a_nx^n$，使其对所有给定 x 所计算出的 $f(x)$与实际对应的 y 值的差的平方和最小，也就是计算多项式的各项系数 $a_1,a_2,\cdots,a_n$。这样一来，$y=f(x)$的曲线虽然不能够精确经过所有的数据点，但也可以给出相对最为近似的曲线。然后再根据得到的多项式函数计算接下来几个数据点对应的函数值来作为预测的数据。

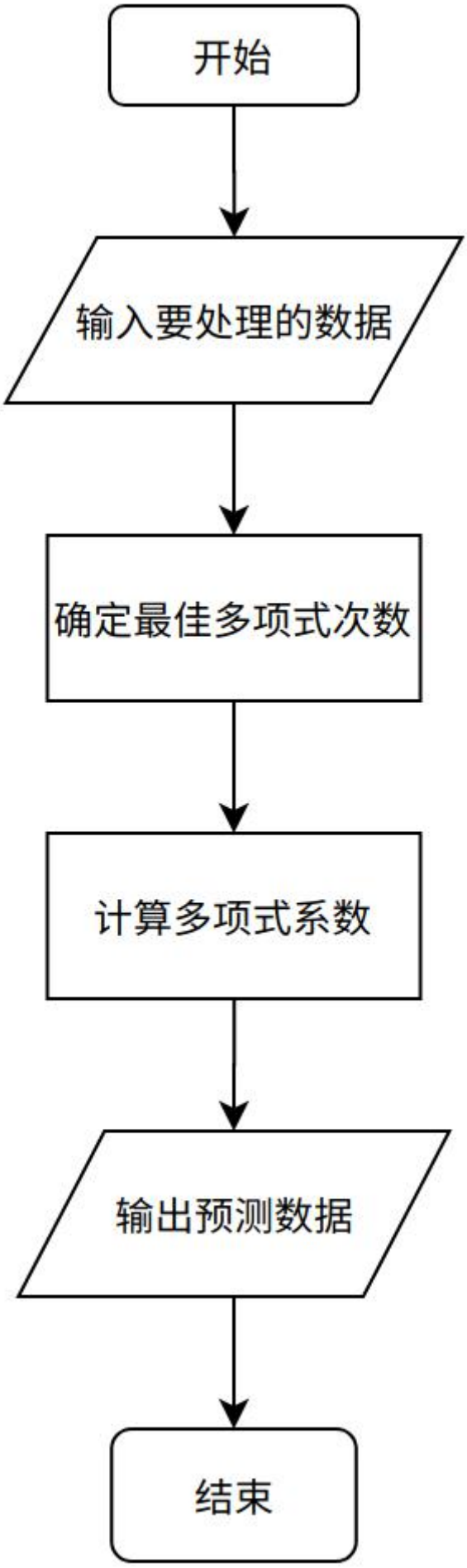
根据最小二乘法的原理，该问题可转换为求以下线性方程组的解。

$$\begin{pmatrix} m & \sum_{i=1}^{m} x_i & \cdots & \sum_{i=1}^{m} x_i^n \\ \sum_{i=1}^{m} x_i & \sum_{i=1}^{m} x_i^2 & \cdots & \sum_{i=1}^{m} x_i^{n+1} \\ \cdots & \cdots & \cdots & \cdots \\ \sum_{i=1}^{m} x_i^n & \sum_{i=1}^{m} x_i^{n+1} & \cdots & \sum_{i=1}^{m} x_i^{2n} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{m} y_i \\ \sum_{i=1}^{m} x_i y_i \\ \vdots \\ \sum_{i=1}^{m} x_i^n y_i \end{pmatrix}$$

在确定线性方程组的各个系数时，我们只需对给定的 $(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)$做相应的计算即可。而解线性方程组相对比较复杂，用到了高斯消元法。

此外，为了让拟合效果更好，我们还运用了“交叉验证”的思想来确定多项式函数最佳的最高次数。即将要处理的数据划分为两部分，前半部分为训练集，后半部分为评估集。用训练集对每个次数的多项式进行拟合，再用拟合后的多项式进行数据预测。然后将预测数据与评估集中的数据进行计算，得到预测误差并计算评估结果。最后误差最小的评估结果所对应的次数即为我们要找的最佳多项式次数。

具体流程图如下：


开始
输入要处理的数据
确定最佳多项式次数
计算多项式系数
输出预测数据
结束

5.2 MQTT 协议

5.2.1 MQTT 原理

MQTT（Message Queuing Telemetry Transport，消息队列遥测传输协议），是一种基于 / （publish/subscribe）模式的“轻量级”通讯协议。该协议构建于 TCP/IP 协议上，由 IBM 在 1999 年发布。MQTT 最大优点在于，用极少的代码和有限的带宽，为连接远程设备提供实时可靠的消息服务。作为一种低开销、低带宽占用的即时通讯协议，使其在物联网、小型设备、移动应用等方面有较广泛的应用。

我们本次仿照物联网应用中对于 MQTT 的使用实现了用户订阅并自动获取通知邮件的功能，以及发布端（发布者 publisher）通过 MQTT 协议（MQTT 服务器）将订阅消息发送至接收端（订阅者 subscriber），接收端将接收到的数据保存在本地文件中。接口类通过读取文件内容，将数据发送至 Vue 前端进行可视化的展示。

5.2.2 MQTT 实现

我们使用 mosquitto+ngrok 在树莓派上搭建了一个 MQTT 服务器，使用 SPRINGBOOT-MQTT 进行 MQTT 服务器的链接，其中发布者负责发布疫情信息，一个接收者负责接收疫情信息后向已经订阅的用户邮箱中发送邮件通知用户疫情信息。