**土壤监测系统开发⽂档**

**1. 引⾔**

1.1 ⽬的

本开发⽂档旨在详细描述土壤监测反应系统的设计和实现，以指导开发团队的开发⼯作，确保系统的功能和性能达到预期⽬标。

1.2 范围

⽂档涵盖了系统的架构设计、数据库设计、模块设计、接⼝设计以及开发环境配置等内容。

**2. 系统架构**

2.1 概述

这个系统的架构图包含了以下几个主要部分:

1，土壤传感器:

• 负责采集土壤湿度、温度、pH 等数据。

2，传感器节点:

• 接收来自传感器的数据,并将其传输到云平台。

3，云平台:

• 接收并存储来自传感器节点的数据。

• 提供数据分析功能,生成可视化报告。

• 通过用户界面展示数据和分析结果。

4，数据库:

• 存储从传感器节点收集的原始数据。

5，分析模块:

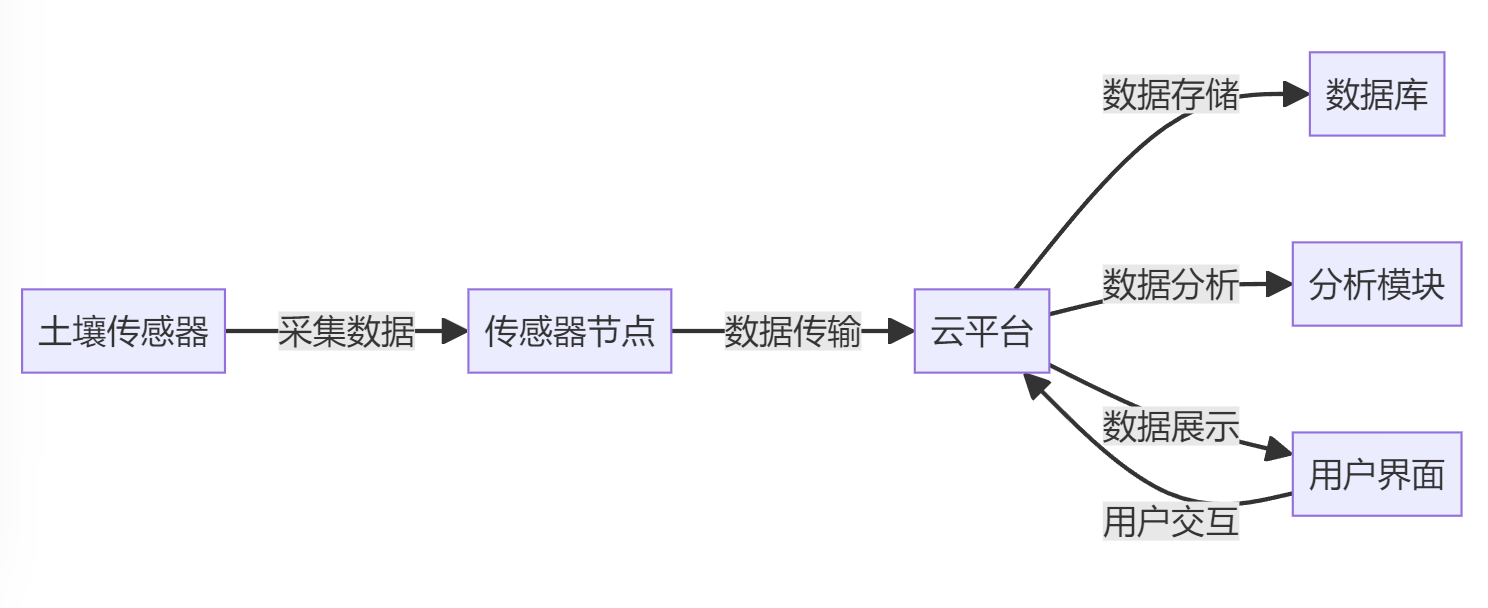
• 对存储在数据库中的数据进行分析和处理，生成有价值的信息。

6，用户界面:

• 提供给用户查看和交互数据的入口,包括移动应用程序或 Web 门户。

• 用户可以通过界面设置监测阈值、查看报警信息等。

2.2 架构图



2.3 技术栈

前端：HTML, CSS, JavaScript, React

后端：Java, Spring Boot

数据库：MySQL

其他：Nginx, Docker

**3. 数据库设计**

3.1 数据库概述

系统使⽤MySQL作为数据库，包含传感器数据表、设备信息表、报警信息表和用户表。

3.2 数据库表设计

传感器数据表(sensor)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 约束 | 描述 |
| device\_id | VARCHAR | 50 | 主键 | 设备ID |
| soil\_moisture | FLOAT | 10 |  | 土壤湿度 |
| soil\_temperture | FLOAT | 10 |  | 土壤温度 |
| soil\_ph | FLOAT | 10 |  | 土壤pH值 |
| light\_intensity | FLOAT | 10 |  | 光照 |
| timestamp | TIMESTAMP |  |  | 数据采集时间戳 |
| battery\_level | FLOAT | 10 |  | 设备电池电量 |

设备信息表(device)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 约束 | 描述 |
| device\_id | VARCHAR | 50 | 主键 | 设备id |
| location | VARCHAR | 100 |  | 设备位置 |
| status | VARCHAR | 20 |  | 设备状态 |
| last\_update\_time | TIMESTAMP |  |  | 最后更新时间 |
| battery\_level | FLOAT | 10 |  | 设备电池电量 |

报警信息表(alert)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 约束 | 描述 |
| alert\_id | INT | AUTO\_INCREMENT | 主键 | 报警ID |
| device\_id | VARCHAR | 50 |  | 设备ID |
| alert\_type | VARCHAR | 50 |  | 报警类型 |
| alert\_level | INT | 3 |  | 报警等级 |
| alert\_time | TIMESTAMP |  |  | 报警时间 |
| alert\_status | VARCHAR | 20 |  | 报警状态 |

用户表(user)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 约束 | 描述 |
| user\_id | INT | 11 | 主键 | 用户ID |
| username | VARCHAR | 50 |  | 用户名 |
| password | VARCHAR | 50 |  | 密码 |
| email | VARCHAR | 100 |  | 邮箱 |

**4. 模块设计**

4.1 传感器节点模块

数据采集：通过土壤湿度、温度、pH等传感器定期采集土壤状态数据。

数据处理：由微控制器对采集的原始数据进行格式转换、异常值检测等初步处理。

数据传输：利用无线通信模块(如WiFi、蓝牙、LoRa)将处理后的数据发送到通信网关。

供电管理：对电池等供电设备进行电量和充电状态的管理。

4.2 通信网关模块:

数据接收：接收来自传感器节点的土壤监测数据。

数据预处理：对接收的数据进行聚合和进一步的格式转换、异常值过滤等预处理。

数据上传：将预处理后的数据上传到云平台。

设备管理：管理连接的传感器节点设备。

4.3云平台模块:

数据接收：通过API接口接收来自通信网关的土壤监测数据。

数据存储：将接收到的数据永久存储到数据库,支持长期数据保存。

数据分析：对存储的数据进行统计分析、异常检测、趋势预测等,生成可视化报告。

报警管理：根据用户设置的阈值触发相应的报警通知,如邮件、短信等。

用户管理：提供用户注册、登录、权限管理等功能,支持多用户协作。

API服务：通过标准化的API为移动应用程序或Web门户提供数据访问和系统功能。

4.4 移动应用/Web 门户模块:

数据展示：以图表、仪表盘等形式直观地显示土壤监测数据。

报警通知：接收来自云平台的报警信息,并通过推送通知用户。

设置管理：允许用户配置监测阈值、添加/管理监测点等。

数据分析：提供对监测数据的深入分析和可视化功能,如趋势分析、异常检测等。

4.5 系统管理模块

设备管理：添加、删除和管理传感器节点设备,查看设备状态。

用户管理：创建、编辑和删除系统用户,分配不同的访问权限。

系统监控：监控云平台的运行状况,如服务状态、资源利用率、错误日志等。

系统配置：管理系统的全局设置,如报警规则、数据保留策略、通知设置等。

**5. 接⼝设计**

5.1 RESTful API

⽤户注册

URL：/api/register

⽅法：POST

请求参数：json

{

"username": "string",

"password": "string",

"email": "string"

}

响应：json

{

"success": true,

"message": "User registered successfully."

}

⽤户登录

URL：/api/login

⽅法：POST

请求参数：json

{

"username": "string",

}

"password": "string"

响应：json

{

"success": true,

"token": "jwt-token"

}

**6. 开发环境配置**

6.1 前端环境

Node.js：版本12.x或以上

React：版本17.x或以上

包管理⼯具：npm或yarn

6.2 后端环境

Java：版本11或以上

Spring Boot：版本2.5.x

构建⼯具：Maven或Gradle

6.3 数据库

MySQL：版本8.0或以上

6.4 其他⼯具

Docker：⽤于容器化部署

Nginx：⽤于反向代理和负载均衡

**7. 部署**

7.1 部署步骤

克隆代码库到服务器。

配置数据库连接信息。

使⽤Maven/Gradle构建后端代码。

使⽤npm/yarn构建前端代码。

配置Nginx，将前端静态资源和后端API进⾏代理。

使⽤Docker容器化部署前后端服务。

7.2 注意事项

确保服务器的安全设置，防⽌未经授权的访问。

定期备份数据库，以防数据丢失。

**8. 测试**

8.1 测试策略

单元测试：针对每个模块的功能进⾏单元测试，确保各功

能模块的正确性。

集成测试：测试模块间的集成接⼝，确保系统各部分协同

⼯作。

⽤户验收测试：邀请实际⽤户进⾏系统验收测试，确保系

统满⾜⽤户需求。

8.2 测试⼯具

JUnit：⽤于Java后端的单元测试。

Mockito：⽤于模拟依赖和进⾏单元测试。

Selenium：⽤于前端的⾃动化测试。