智慧城市管理平台 (Smart City Platform) - 技术开发手册

版本号: v1.2.0
编写日期: 2025-12-07

1. 详细设计说明 (Detailed Design)

1.1 系统架构

本项目采用 **B/S** 架构，前后端分离开发模式。

- 前端: Vue 3 + Vite + Cesium (3D GIS) + ECharts (数据可视化) + Pinia (状态管理)。
- 后端: Spring Boot 2.x/3.x + MyBatis + MySQL。
- 数据交互: RESTful API，使用 Axios 进行 HTTP 请求。

1.2 核心业务流程 (Sequence Diagram)

以下是设备数据模拟上报及前端实时展示的处理流程：

```
sequenceDiagram
    participant ScheduledTask as 后端定时任务 (DeviceTask)
    participant Service as 业务逻辑层 (Service)
    participant DB as 数据库 (MySQL)
    participant API as 前端接口 (API)
    participant Dashboard as 前端大屏 (Vue)

    Note over ScheduledTask: 每隔X秒触发
    ScheduledTask->>DB: 查询所有在线设备
    DB-->>ScheduledTask: 返回设备列表
    loop 遍历设备
        ScheduledTask->>ScheduledTask: 生成随机模拟数据(温湿度)
        ScheduledTask->>DB: 插入 iot_device_history
    end

    Note over Dashboard: 前端轮询/刷新
    Dashboard->>API: GET /system/stats
    API->>Service: 获取最新统计
    Service->>DB: 聚合查询今日数据
    DB-->>Service: 返回统计结果
    Service-->>API: 返回 JSON
    API-->>Dashboard: 更新图表组件
```

1.3 模块划分

1. **3D可视化中心 (Cesium Container)**:
    - 负责加载城市3D模型/瓦片数据。
    - 展示设备点位（POI）及状态弹窗。
2. 数据驾驶舱 **(Dashboard)**:
    - 头部 **(Header)**: 系统标题、天气、时间。
    - 左侧面板 **(DataPanelLeft)**: 城市概览数据、环境监测图表。
    - 右侧面板 **(DataPanelRight)**: 实时告警信息、设备状态分布。
3. 后台管理系统:
    - 设备管理: 设备的增删改查 (CRUD)。
    - 用户/部门管理: 组织架构及权限基础。
    - 日志系统: 记录操作日志。
4. 模拟仿真模块:
    - 后端定时任务模拟IoT设备数据上报。

2. 数据库设计说明 (Database Design)

2.1 E-R 图 (Entity-Relationship Diagram)

```
erDiagram
    SYS_DEPT ||--o{ SYS_USER : "has members"
    SYS_DEPT {
        bigint dept_id PK
        string dept_name
        bigint parent_id
    }
    SYS_USER {
        bigint user_id PK
        string username
        bigint dept_id FK
    }
    IOT_DEVICE ||--o{ IOT_DEVICE_HISTORY : "generates"
    IOT_DEVICE {
        bigint device_id PK
        string device_name
        string device_type
        string location
        char status
    }
    IOT_DEVICE_HISTORY {
        bigint history_id PK
        bigint device_id FK
        string value
```

```
    datetime create_time
}
SYS_LOG {
    bigint log_id PK
    string content
}
```

2.2 数据表结构

2.2.1 用户表 (sys_user)

字段名	类型	说明	主键/外键
user_id	BIGINT	用户ID	PK
dept_id	BIGINT	部门ID	FK
user_name	VARCHAR	用户名	
password	VARCHAR	密码	
email	VARCHAR	邮箱	
phonenumber	VARCHAR	手机号	
status	CHAR	状态(0正常 1停用)	

2.2.2 部门表 (sys_dept)

字段名	类型	说明	主键/外键
dept_id	BIGINT	部门ID	PK
parent_id	BIGINT	父部门ID	
dept_name	VARCHAR	部门名称	
order_num	INT	显示顺序	

2.2.3 物联网设备表 (iot_device)

字段名	类型	说明	主键/外键
device_id	BIGINT	设备ID	PK
device_name	VARCHAR	设备名称	
device_type	VARCHAR	类型(监控/传感器等)	
location	VARCHAR	经纬度坐标 (JSON或String)	
status	CHAR	状态(0在线 1离线)	
last_active_time	DATETIME	最后活跃时间	

2.2.4 设备历史数据表 (iot_device_history)

字段名	类型	说明	主键/外键
history_id	BIGINT	记录ID	PK
device_id	BIGINT	设备ID	FK
value	VARCHAR	数据值(温度/湿度等)	
create_time	DATETIME	采集时间	

2.3 初始化SQL脚本

(请参考 v1.1.0 版本或直接执行提供的 database demo/新建 文本文档.txt)

3. 接口文档 (API Documentation)

基础路径: /api

3.1 通用状态码

Code	Message	Description
200	Success	请求成功

401	Unauthorized	Token无效或过期
403	Forbidden	无权限访问
500	Error	系统内部错误

3.2 核心接口定义

3.2.1 认证模块 (AuthController)

- **POST /login**
  - 入参: { "username": "admin", "password": "..." }
  - 出参: { "code": 200, "token": "jwt-token-string" }

3.2.2 设备管理 (DeviceController)

- **GET /device/list**
  - 入参: pageNum, pageSize
  - 出参: { "total": 100, "rows": [...] }
- **POST /device**
  - 入参: { "deviceName": "...", "deviceType": "...", "location": "..." }

3.2.3 系统统计 (SystemController)

- **GET /system/stats**
  - 说明: 获取大屏首页的聚合数据。
  - 出参:

```
{
  "totalDevices": 150,
  "onlineRate": 98.5,
  "todayAlerts": 12,
  "environment": { "temperature": 24.5, "humidity": 60 }
}
```

4. UI/UX 设计规范

4.1 色彩系统

- 主色调 (Cyan): #00f2ff - 用于科技线条、选中态。
- 背景色 (Dark): #0b1a2f - 深色空天背景。
- 辅助色: 黄色(#ffcc00, 警告)、红色(#ff3333, 故障)。

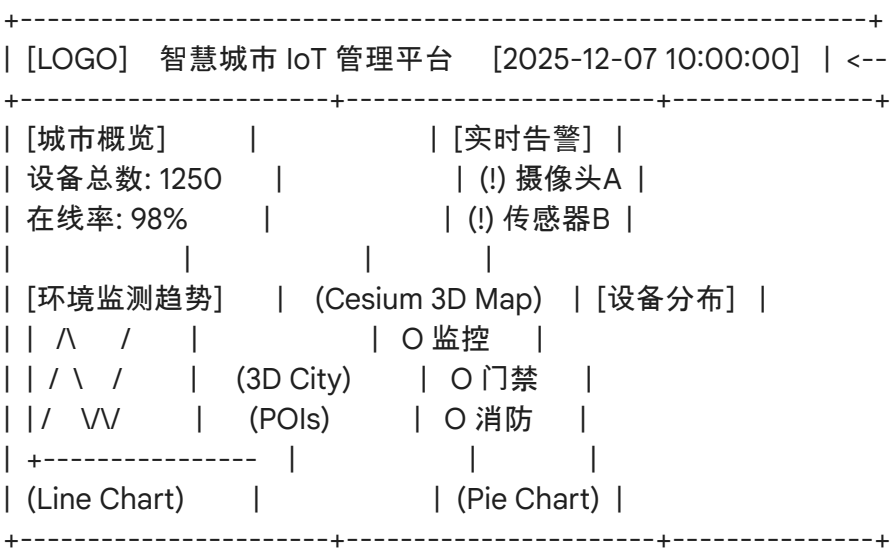
4.2 交互规范

- **Hover**: 鼠标悬停图表时显示 Tooltip。
- **Click**: 点击地图 POI 弹出详情 Modal，背景需做模糊处理 (backdrop-filter: blur(4px))。

5. 产品原型 (Product Prototype)

5.1 数据驾驶舱线框图

使用 ASCII/Mermaid 描述大屏的布局结构：



5.2 页面标注

1. **DashboardHeader**: 高度 80px，绝对定位 top: 0，z-index: 10。
2. **DataPanelLeft/Right**: 宽度 25%，背景使用 rgba(0,0,0,0.6)，带边框发光效果。
3. **CesiumContainer**: 全屏铺满，z-index: 0。

6. 算法设计说明 (Algorithm Design)

6.1 模拟数据生成 (Simulation)

由于没有真实传感器，系统使用高斯分布模拟环境数据波动：
$$ T_{current} = T_{base} + \sigma \cdot Random() $$
其中 $T_{base}=25$ (基准温度)，$\sigma=2.0$ (标准差)。

6.2 坐标转换

Cesium 使用 WGS84 笛卡尔坐标系，MySQL 存储经纬度。
转换公式：
Cartesian3 = Cesium.Cartesian3.fromDegrees(longitude, latitude, height);