实验3：电梯传感器数据监控系统 - 后端实验设计

基于实验2已完成的Vue.js前端项目，设计配套后端实现（Python/Java可选）。

1、目的和要求

1. 掌握后端API设计与实现
2. 掌握使用 `Python(Flask/Django)` 或 `Java(Spring Boot)` 构建RESTful API
3. 理解前后端分离架构的数据交互
4. 掌握数据库设计与CRUD操作
5. 实现JWT认证机制

2、技术栈选项（二选一）

2.1 Python版本

- **框架**: Flask/Django
- **数据库**: SQLite/MySQL
- **ORM**: SQLAlchemy/Django ORM
- **认证**: PyJWT

2.2 Java版本

- **框架**: Spring Boot
- **数据库**: MySQL/H2
- **ORM**: Spring Data JPA/MyBatis
- **认证**: Spring Security + JJWT

3、核心API设计

功能	方法	端点	描述
用户登录	POST	`/api/auth/login`	返回JWT token
电梯列表	GET	`/api/elevators`	支持状态/时间筛选
电梯详情	GET	`/api/elevators/{id}`	包含基本信息+状态
传感器数据	GET	`/api/elevators/{id}/sensors`	返回当前传感器数据（含异常标记）

4、数据库参考

具体数据库管理系统可自选，入mysql、sqlite等

```
USERS {
    int id PK
    varchar(50) username
    varchar(100) password
    varchar(20) role
    timestamp created_at
}

ELEVATORS {
    int id PK
    varchar(50) name
    varchar(100) location
    enum("normal","warning","fault") status
    date last_maintenance
}

SENSORS {
    int id PK
    int elevator_id FK
    varchar(30) type
    float max_value
    float min_value
}
```

```
SENSOR_DATA {
    int id PK
    int sensor_id FK
    float value
    bool is_abnormal
    timestamp timestamp
}
```

5、关键实现步骤参考

1、项目初始化

```
Python
pip install flask flask-sqlalchemy pyjwt

Java
spring init --dependencies=web,jpa,security,lombok
```

2、JWT认证示例（Python Flask）

```
from flask_jwt_extended import create_access_token
@app.route('/login', methods=['POST'])
def login():
   username = request.json.get('username')
```

6、实验报告要求

1. **原理分析**
    - RESTful设计规范
    - JWT令牌的组成与验证流程
2. **实现过程**
    - 数据库连接配置
    - 认证模块实现关键代码
    - 异常处理机制
3. **测试结果**

- Postman测试截图（含请求/响应）
- 前端集成效果图

7、提交材料

1. 实验报告（PDF格式）
2. 源代码
3. `postman_collection.json` 测试用例
4. 系统架构图（可手绘拍照）

> **提示**：前端项目需修改 `axios` 基地址为 `http://localhost:8081/api`（Java）或 `http://localhost:5000/api` （Python