从 PLC 到 PC 控制的第一步

前后端架构与 REST API 初体验

讲师：[May 周 晓婷] | 体验课时长：25 分钟

Step One: From PLC to PC

Understanding Backend, Frontend, and REST API

Instructor: [May Xiaoting Zhou] | Trial Class Duration: 25 Minutes

📚 课程预览 | Class Outline

👋 学生背景访谈 & 技术词汇确认

Interview: Your background & vocabulary alignment

📊 PLC vs PC 控制架构图解

Architecture: PLC vs PC overview

🧪 REST API 实作演示（FastAPI）

Hands-on: Build your first control API

🔁 回顾解疑

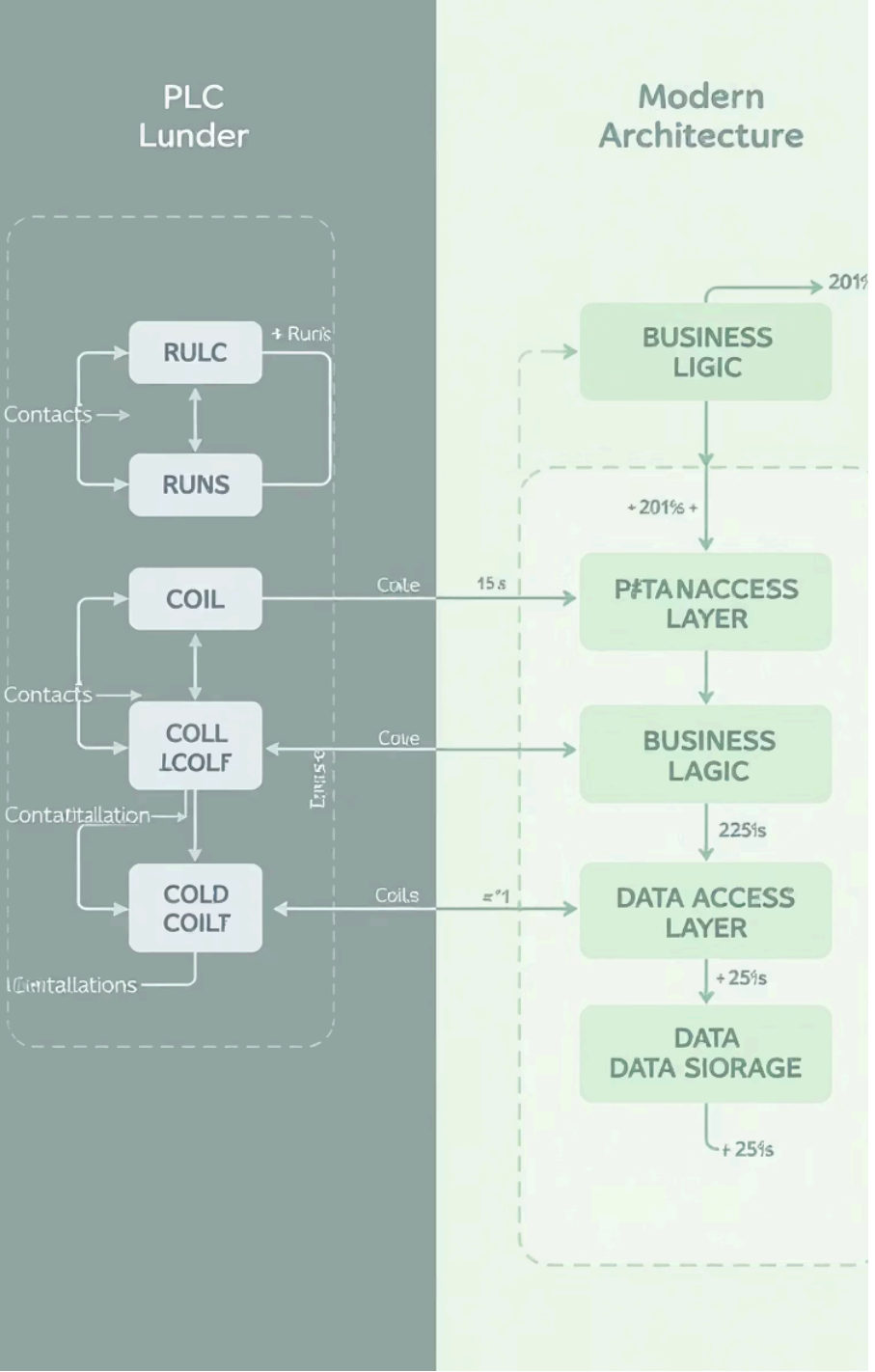
Review & QA

📅 课程安排和计划

Next steps: Course plan & future schedule

🧠 学生背景确认 + 技术词汇对齐

- 你熟悉的PLC流程长什么样?
  - What's your current PLC control workflow like?
- 听过 API / 前端 / 后端 这些术语吗?
  - Have you heard terms like API, Frontend, Backend?
- 知道 Python 函数和变量的意思吗?
  - Are you familiar with Python functions or variables



PLC vs PC 控制系统架构

PLC 控制系统

- 程序流程图逻辑
- 中心化逻辑
- 输入输出直接连线

Logic flows in diagrams

PC 控制系统

- 软件架构分层
- 前端 / 后端 / 数据库
- 使用 API 做模块通信

Layered system — UI, Backend, Database

💡 API = 模块间的共享记忆，实现系统各部分的高效通信

API = Like shared memory for module communications

💻 REST API 实作演示（FastAPI）

```
from fastapi import FastAPI
app = FastAPI()
device_status = {"pump": "OFF"}

@app.get("/status")
def get_status():
    return device_status

@app.post("/toggle/{device}")
def toggle_device(device: str):
    if device in device_status:
        device_status[device] = "ON" if device_status[device] == "OFF" else "OFF"
    return device_status
```

API 功能说明：

- .get() = 读取状态
- .post() = 切换设备（ON/OFF）
- Swagger UI 可立即看到状态切换
- Use Swagger UI to simulate a button controlling device logic.
- This simple API already demonstrates the core principles of modern control systems.

🔁 今天你学到了什么?

1. 你学到哪些和 PLC 不一样的概念?

   What's different from PLC logic?

2. API 是什么？为什么它很重要?

   Why are APIs important in modern control systems?

3. 想象你要用这个系统控制什么设备?

   What device would you apply this system to?

📅 课程安排和计划

🎓 课程目标

- 建立从PLC到PC控制系统的架构思维
- 理解前后端协作、API通信、数据库整合
- 掌握Python + Vue + SQL + Modbus的基础整合开发能力
- 能实际做出一个具备控制、显示、记录的Web控制系统

🗂️ 10节课程 + 课后项目（每周1课）

第1课：PC控制 vs PLC控制架构总览 + Python快速回顾

内容：

- PLC vs PC 控制模型比较（逻辑、设备、架构）
- PC端开发工具介绍（VS Code, Python环境）
- Python语法复习：变量、条件判断、循环、函数

课后项目：

- 使用 Python 实作一个 LED 闪烁模拟器（透过 print 模拟 ON/OFF 循环控制）

第2课：前端/后端/数据库三层架构 + API基础

内容：

- 什么是前后端架构？数据如何流动？
- HTTP、请求方法（GET/POST）、JSON数据格式
- RESTful API的核心概念

课后项目：

- 用 Postman 模拟一组 API 请求（GET/POST 控制灯号状态）

第3课：后端实作入门：用 FastAPI 写控制接口

内容：

- 建立一个 FastAPI 项目
- 编写控制灯号的接口（模拟输出控制）
- 使用 Swagger 文档测试 API

课后项目：

- 实作 API：
  - GET /status 获取状态
  - POST /toggle 切换状态

第4课：前端UI入门（HTML/CSS/JS/Vue）

内容：

- 建立基本网页结构
- 使用 Vue 绑定数据、按钮控制
- 简易状态显示与切换按钮

课后项目：

- 建立网页控制面板，能切换“装置开关”状态（使用按钮切换文字）

第5课：SQL数据库与Python存取（SQLite）

内容：

- 数据表设计（装置状态、历史记录）
- SQLite 基础语法（SELECT、INSERT、UPDATE）
- Python + sqlite3 读写数据库

课后项目：

- 建立一张状态记录表，每次切换设备状态时写入数据库

第6课：前后端整合：用Axios调用API

内容：

- 前端使用 Axios 发送请求（GET/POST）
- UI 显示设备当前状态 / 控制设备
- 处理 JSON 响应、更新前端显示

课后项目：

- 完成完整流程：Vue 页面可点按钮 → 控制设备 → 显示状态更新

第7课：Modbus通讯基础与Python实作

内容：

- Modbus TCP/RTU协议介绍
- 使用 pymodbus 实作 PC 读取/写入 模拟设备
- 设计指令映射表（设备编号、寄存器说明）

课后项目：

- 用 Python 实作读取设备温度值（用随机值模拟或搭配Modbus模拟器）

第8课：状态机设计与Python逻辑控制

内容：

- 状态机（State Machine）介绍
- 用 Python 模拟阶梯图的顺序控制逻辑
- 多设备控制流程设计

课后项目：

- 设计一个三步顺序控制逻辑（如：A设备启动→延迟→B设备运行→完成）

第9课：整合项目开发实战：小型工业Web控制系统

内容：

- 整合前端 + 后端 + Modbus + 数据库
- 构建真实控制流程（输入、反馈、记录）
- 系统调试、错误处理、日志打印

课后项目：

- 实作完整系统：
  - 控制灯号/马达模拟
  - 读取温度并显示在网页
  - 所有操作记录进数据库

第10课：系统部署与扩展方向

内容：

- 部署方法介绍（Uvicorn / Docker基础）
- 系统维护与扩展建议（多个装置、权限控管）
- 后续学习路径建议（MQTT、OPC UA、Grafana等）

课后项目：

- 将系统部署到本机服务中，并撰写一份简单的用户手册说明书

📚 接下来的课程你将会...

1. 第3周：网页控制真实设备

   Week 3: Control real devices through web interface

2. 第7周：整合 Modbus 通讯

   Week 7: Integrate Modbus communications

3. 第9周：打造完整前后端控制系统

   Week 9: Build a complete full-stack control system

> 💡 你已具备扎实的控制逻辑思维，只需掌握系统架构、工具应用与实作方法，就能顺利实现从 PLC 到 PC 的技术转型。

With your strong control logic mindset, mastering system architecture and practical tools is all it takes to make a smooth leap from PLC to modern PC-based control.

课程总结 | Class Summary

PC控制优势

现代化架构允许更灵活的系统设计和扩展

API的核心作用

简化模块间通信，标准化数据交换

学习路径

从PLC基础到全栈控制系统的进阶之旅

分层架构思维

前端、后端、数据层的清晰分工

感谢参与今天的体验课！欢迎在下课后提出任何问题或分享你的想法。

Thank you for joining today's trial class! Feel free to ask any questions or share your thoughts after class.