CASE STUDY / IOT / 2026

技术验证 · 未上线　匿名案例

设备没有屏幕，手机就是唯一入口。

一个 BMS 计量 IoT 项目的全栈技术验证：微信小程序 × BLE 设备协议 × Java 后端 × OTA。

PROJECT SNAPSHOT　VALIDATED

24 后端端点	**15** 数据库表
3 设备模式	**17** Node 测试套件

手机负责设备控制；后端负责账号、归属、密钥协商与记录。

没有硬件，也能把端到端链路讲清楚。

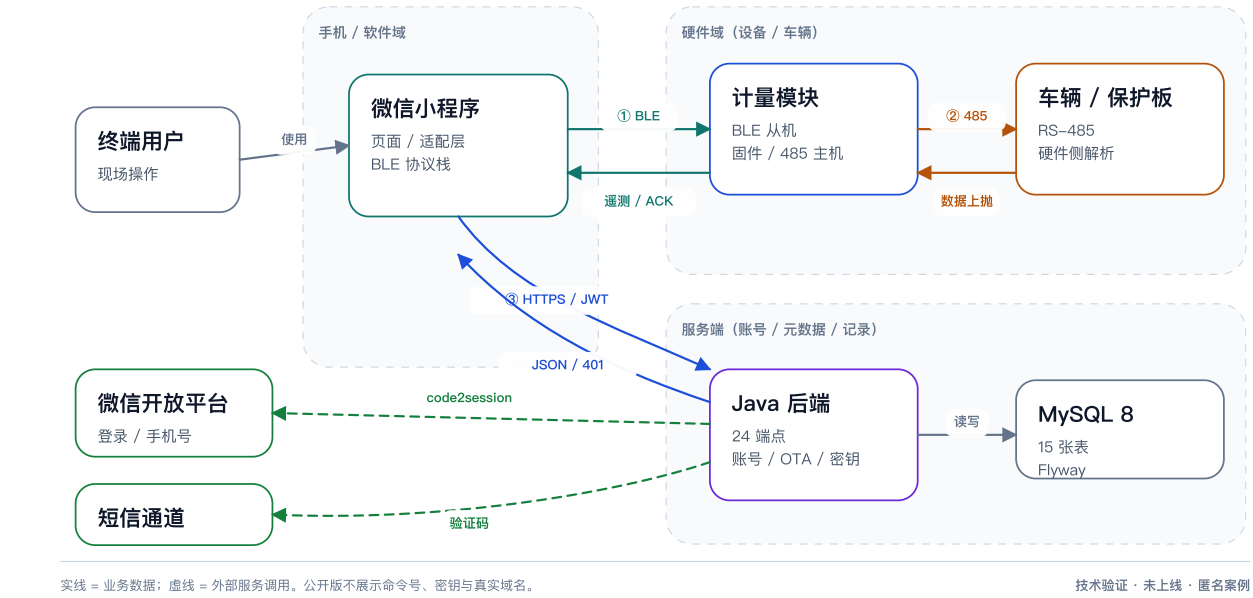
这不是一个“已经上线服务客户”的生产案例。

“它是一份技术验证：把设备、协议、后端、测试和边界，先跑通，再讲清楚。”

01

先看系统，不看代码

手机是唯一人机入口；后端不直连设备；设备控制发生在手机与计量模块之间。



系统上下文。公开版不展示命令号、密钥与真实域名。

01

手机承担交互

设备没有屏幕，状态、操作、错误提示都在小程序里完成。

02

后端不碰硬件

账号、绑定、密钥协商、OTA 元数据与记录，才是后端职责。

03

硬件问题不背锅

软件边界写清楚：不承诺硬件与固件行为。

02

三条链路，三种责任

把“谁和谁说话、谁负责什么”拆开，后面的协议、接口和验收才有落点。

① BLE

手机 ↔ 计量模块

握手、轮询、参数读写、OTA。软件域里最“像硬件”的一段。

小程序 → BLE 帧 → 计量模块

② RS-485

计量模块 ↔ 车辆/保护板

硬件域内部通信。软件只消费硬件上抛的结果，不替硬件做承诺。

计量模块 → 485 总线 → 保护板

③ HTTPS / JWT

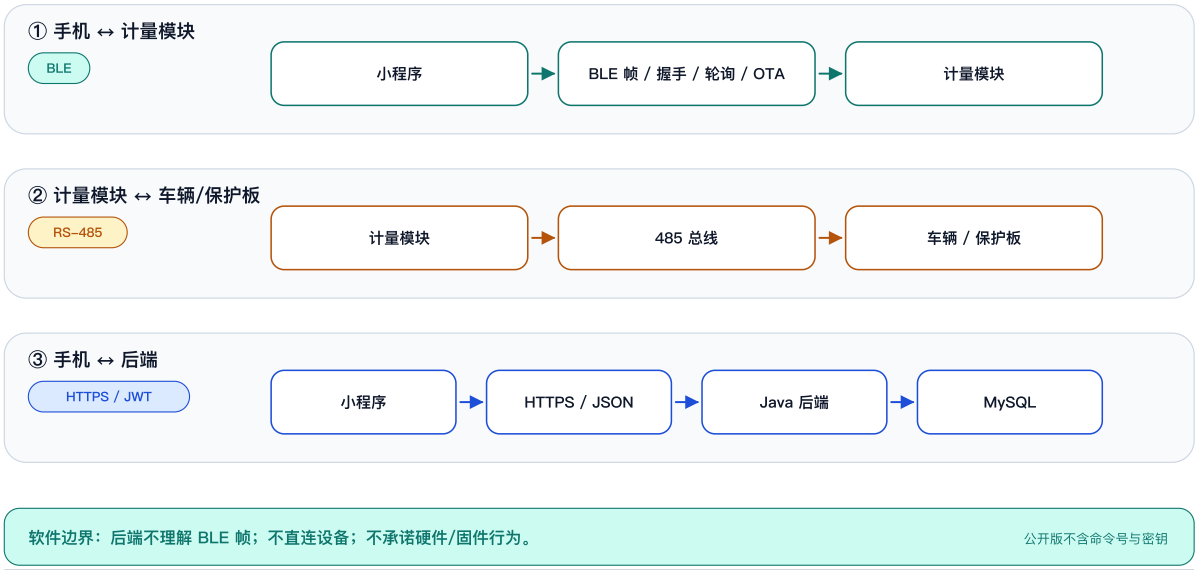
手机 ↔ 后端

登录、设备归属、密钥协商、OTA 元数据、遥测记录。后端从不直连设备。

小程序 → Java 后端 → MySQL

三条链路与职责边界

手机负责设备控制；后端负责账号、归属、密钥协商与记录



软件边界：后端不理解 BLE 帧；不直连设备；不承诺硬件/固件行为。 公开版不含命令号与密钥

链路 ② 属于硬件域；软件只消费硬件上抛的结果。 技术验证 · 未上线 · 匿名案例

三条链路。链路 ② 属于硬件域；软件只消费结果。

03

五个关键决策

真正决定项目能不能讲清楚的，不是用了多少技术，而是边界有没有提前写下来。

01

手机做协议栈

设备无屏，手机就是唯一入口。BLE 帧、握手、轮询、OTA 都放在小程序侧编排。

02

后端不直连设备

后端只做账号、归属、密钥协商与记录，避免把硬件风险拉进服务端。

03

密钥协商分两段

临时公钥 + 设备公钥，手机转发，后端计算 shared key；设备控制仍在手机侧。

04

OTA 以版本核验收尾

文件发出去不算成功；重启、重连、版本匹配、记录可查，才算闭环。

05

测试分三层

Node 契约/离线测试、Java 单测、真机联调。每层能证明什么，提前写清楚。

04

OTA 不是传文件

可校验、可恢复、可追责，才叫 OTA 闭环。无法重连核验 ≠ 成功。



OTA 证据链。公开版不展示固件、哈希、设备标识与真实升级记录。

> **设备重启后没有重连、没有版本核验，就不能写“升级成功”。**

05

后端只做该做的事

轻量分层，AppService 直接调 Mapper。层数少、依赖少，适合 4.4k 行规模。

24 个端点	5 个 Controller + Health
8 个 AppService	账号、绑定、密钥、OTA、SOC、短信等

15 张表	MySQL 8 + Flyway
一个取舍	不引入 Repository 层；实体可能穿越 web 层，改表可能影响 API 契约

这不是大型企业级分层模板，而是一个 4.4k 行规模下“够用、能讲清楚”的选择。



Java 后端分层。

06

握手：后端从未直连设备

公钥交换与归属校验由手机转发；后端只计算 shared key，设备控制仍在手机侧。



BLE 握手时序（高层）。公开版不含命令号、字节序、密钥派生与真实抓包。

07

测试与证据

数字不是装饰。每个数字都要能说清楚口径、环境和边界。

24 后端端点	**15** 数据库表	**16** 小程序页面
3 设备模式	**17** Node 测试套件	**69** Java 单测（归档）

Node 测试

16/17 套离线通过；`java-contract` 需后端 + MySQL，按设计跳过。15 套计数合计 187 个用例，另有 `offline-demo` 通过。

Java 单测

案例归档口径为 69/69（JDK 17）。2026-10-20 在当前仓库复跑：OpenJDK 17.0.20，**155/155 通过**，含归档后新增的管理后台测试；不再把 JDK 24 的 Mockito 兼容性失败当作当前结论。

静态自检

0 错 0 警。测试、文档、脚本都保留在仓库里，而不是只存在于演示视频里。

08

边界先说清楚

技术验证完成，不等于生产上线。这句话必须写在案例前面。

已完成

技术验证、端到端演示、测试与文档沉淀。

未完成

签约、上线、真实硬件适配、规模化运维。

公开材料不含

客户信息、真实 appid、密钥、固件、协议细节。

这是一个可复用的 PoC 样板，不是“已服务客户”的生产案例。

09

如果重来一次

项目终止不等于工作没有价值。至少有三件事，下次可以更早做。

01

先签 PoC 协议

再写第一行代码。先做再签，最后卡在合同上，对双方都不划算。

02

先定验收口径

再定功能清单。什么叫“跑通”，要在开始前写成可检查的条目。

03

先确认软硬件边界

再承诺工期。硬件、固件、协议、云服务，各自的责任要落在纸面上。

NEXT STEP

你也在做“设备 + 小程序 + 后端”的项目?

先花 30 分钟，把边界、链路和验收口径讲清楚。再决定要不要做。

预约一次项目评估