交付 / 售前 / 解决方案 · 一天 6 小时

WorkBuddy FDE 交付训练营

从「会用工具」到「能独立交付」

六项特殊技能、一整套 Skill 与 MCP 工具箱、4 次现场演练与 4 张任务卡。离开时你手上应该有一套能带进客户现场的打法。

6 项特殊技能　12 个常用 Skill　自建 MCP 连接器



今天走五个章节

一天 6 小时 · 含 8 次动手

上午讲技能，
下午做交付。

第 2 章讲六项技能的现场做法，每讲完两项就动手一次；第 4 章的四张任务卡分组完成，评分标准写在卡上。

为什么是现在

FDE 岗位近两年需求暴涨，缺的从来不是会写代码的人，而是能把 AI 接进真实业务、并且交付得出去的人。

先修提示：本片默认你已会用 WorkBuddy 交代任务

01 **FDE 是谁** 35 min
三力模型、六项能力、交付闭环，以及三条不能碰的红线
P03–P09 · **六项能力自评表**

02 **六项特殊技能 × WorkBuddy 打法** 90 min
每项技能：现场要做到什么、用哪个 Skill、接哪个 MCP、怎么验收
P10–P22 · **技能-工具映射表 + 4 次演练**

03 **工具箱：Skill 与 MCP 全清单** 60 min
技能怎么装、连接器怎么接、自定义 MCP 怎么写、安全三层
P23–P31 · **个人工具箱清单**

04 **实战演练营** 120 min
四张任务卡：汇报包 / 数据问数 / 知识库智能体 / 集成自动化
P32–P38 · **4 份交付物 + 评审表**

05 **避坑、边界与 90 天** 30 min
七个高频坑、四道安全闸门、从会用到能交付的能力阶梯
P39–P44 · **90 天路线图**

第 1 章 · P03–P09 · 35 分钟

FDE 是谁
从会用工具到能交付

先分清一件事：FDE 不是「更高级的驻场工程师」，
而是能在需求不清、数据不干净、环境受限的情况下，自己把活做完
的人。

本章产出 六项能力自评表 + 三条红线检查动作



FDE 的三力模型

这岗位缺的从来不是会写代码的人

三力必须同时成立。缺任何一条，活都停在「演示很好看，上线没人敢用」。

1 懂技术

AI / 数据 / 系统接口的底层逻辑说得出，能把模型能力翻译成「这个场景能不能做、需要什么条件」。

决定你能不能做

2 懂产业

看得懂客户的流程、口径和考核指标，知道哪一段值得自动化、哪一段绝对不能交给 AI。

决定你做得对不对

3 会落地

在客户的真实环境里把方案跑起来：数据接得上、权限说得清、出问题退得回、人能接手。

决定你能不能交付

三力模型

3

条同时成立

只懂技术，做出来的是自嗨的 demo；只懂产业，写出来的是没人能实现的方案；只会落地不带判断，就是给客户做一个更快的错流程。

自评口径：三项各按 1–5 打分，任一项低于 3 分，先补这项再谈独立交付。

它和隔壁三个岗位差在哪

差别不在技术高低，在于「需求是谁定的」和「结束时人还在不在」

驻场按文档实施，售前按标书交付，产品按共性抽象；FDE 是三者之间那条没人愿意走的路。

角色	需求从哪来	写不写代码	主要交付物	结束时留下什么
传统驻场工程师	客户或合同里写死的需求	写，但改的是既有系统	上线记录、运维报告	系统还在跑，换个人也能接
售前 / 解决方案	自己去找客户，方案要能签单	基本不写，画架构图	方案书、报价、POC 演示	一份看起来能落地的方案
产品经理	从多个客户里提炼共性需求	不写，写 PRD	需求文档、产品路线	一个能卖给很多人的功能
FDE	自己进现场问出来，需求往往是模糊的	写，从零把活跑通	能用的系统 + 交接说明	客户自己能接着往下用

一句话记法：**驻场交付系统，售前交付方案，产品交付功能，FDE 交付「客户自己能接着跑起来的能力」。**

FDE 的交付闭环：五步，每步有时间盒



「FDE 的价值不在最后一次交付，而在每一次交付之后，下一次都能快一点。」

所以第 5 步「反哺」不是可选项，是这份工作能积累下来的唯一方式。

1 勘察 — 1–3 天

进现场看真实流程，把口语需求写成可验收的问题

2 定义 — 半天

选定第一个能跑通的最小切口，写清口径与不做什么

3 构建 — 2–5 天

搭出能用的版本，自己先跑通一遍再给客户看

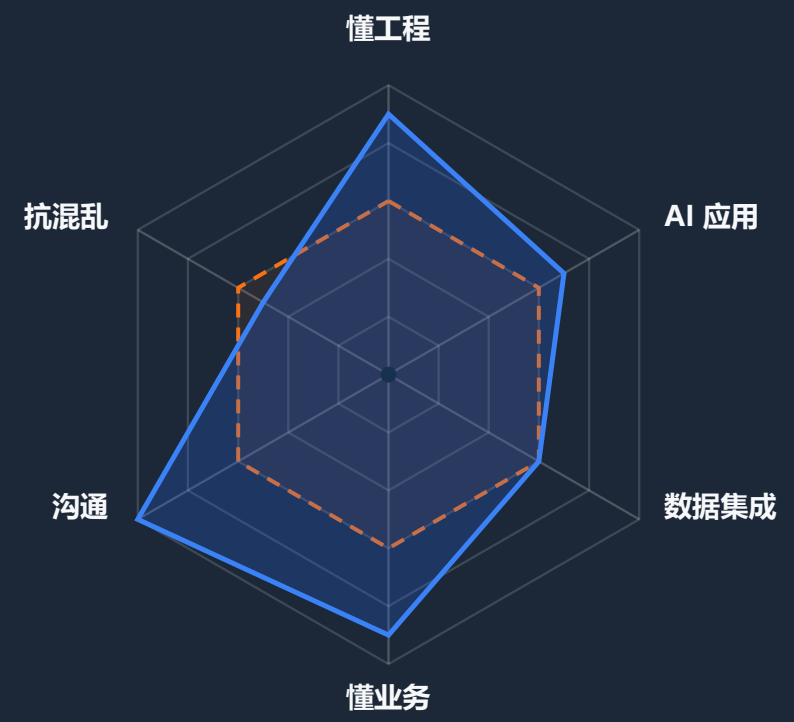
4 上线 — 1 天

装到客户环境里，给权限、给回退、给验收动作

5 反哺 — 持续

把这次的做法封成 Skill 或模板，下次直接复用

六项能力雷达：先知道自己在哪一格



示例学员现状　达标线（每项 3 分）　刻度：中心 0 → 外圈 5

三项及以上低于 3 分，先别接独立交付

雷达是自评工具，不是考核表——它的用途是决定你今天该重点练什么。

懂工程 4.5 / 5

至少熟一门语言，看得懂接口、数据库、日志

AI 应用 3.5 / 5

知道 RAG、智能体、评估分别解决什么问题

数据集成 3 / 5

能接原始数据，自己做清洗与字段映射

懂业务 4.5 / 5

能画出客户流程，说得出哪段不该交给 AI

沟通 5 / 5

同一件事能对老板、IT、一线各讲一遍

抗混乱 2.5 / 5

需求会变、数据会脏，仍然能推进

五个阶段，WorkBuddy 各替你干哪一段

不要把它当成「一个更快的搜索框」，要当成一双手接进你交付流程的五个位置

同一套能力，在不同阶段扮演的角色完全不同：勘察期是眼睛，构建期是手，沉淀期是抽屉。

勘察期 1–3 天	定义期 半天	构建期 2–5 天	交付期 1 天	沉淀期 持续
批量读现场材料 几十份散落的制度、表格、邮件一次性读完，先出目录树再出结论	**先要计划再动手** 计划模式下它先给步骤，你确认之后才开始执行	**写代码与改数据** 从零把活跑通，而不是改别人写好的系统	**生成交付物** 报告、PPT、看板、可访问链接，客户能直接看	**封装成技能** 这次的做法封起来，下次一个词就调用
截图直接解读 报错弹窗、白板照片、纸质表格粘进输入框就行	**写清口径与边界** 输出口径表、不做什么清单、验收动作三件套	**装技能扩能力** 技能市场按需装，缺什么补什么，用完就关	**留验收动作** 客户自己能复现一遍，才算真的交付	**配成自动化** 定时跑，出异常才叫人，平时别打扰
把需求逼具体 用提问模式追问到底，别急着动手	**让专家挑毛病** 换对应行业的专家视角，把方案再审一遍	**接客户系统** 自建 MCP 把内部接口接进来，让它自己查	**写交接说明** 改哪里、怎么改、出错怎么退，一次讲清	**记忆沉淀口径** 客户的口径与偏好不用每次都重讲一遍
提问模式 + 多源读取	计划模式 + 专家	技能 + MCP 连接器	文档技能 + 发布为应用	技能创建 + 自动化 + 记忆

三条红线：看着没问题，出事都是这样出的

红线不是流程要求，是「你已经越界了」的判定线

下面每条都配了一个真实会发生、而且当时感觉完全合理的反例。看到自己中招过，就在这一页多停一会儿。

01 客户数据不出域

✗ 「这份表贴进去让它帮我排一下时间表」

数据一旦离开客户边界，就再也收不回来了——不管你有没有恶意。

✓ 先在脱敏样本上把流程跑通，正式数据只在客户环境内处理。

检查动作 · 动手前问一句：这份数据，允许离开哪里？

02 对外动作必确认

✗ 「让它自动把报告发到客户群里，省一步是一步」

自动化发出去的内容，撤回成本远高于多发一次人工确认。

✓ 生成好、等你确认再发；发送类动作永远保留一个人工确认点。

检查动作 · 凡是会到达他人的动作，先手工确认一次再放开。

03 生产环境最小权限

✗ 「反正是查一下就完事，直接给完全访问省得麻烦」

省下的那点麻烦，会在某次误删或误改时一次性还给你。

✓ 能只读就不给写；能限定目录就不给全盘；先在测试环境验证。

检查动作 · 给权限前先写下：这个任务最少需要什么？多的一律不给。

第 2 章 · P10–P22 · 90 分钟

六项特殊技能
每一项都有工具对应

这一章不讲道理，只讲做法：每项技能写清「现场要做到什么」，再给出对应的 Skill、MCP 和验收动作。讲完两项就动手一次。

本章产出 技能-工具映射表 + 4 次现场演练 + 7 项交付验收清单



六项技能 × WorkBuddy：一张表看完

6 六项里真正稀缺的是第 1 项和第 6 项

技术活能学；把需求问清楚、把成果讲明白，才是分水岭。

Skill 管「怎么做」　MCP 管「连到哪」

#	技能	现场要做到什么	主力 Skill	主力 MCP / 连接器
1	**现场勘察与需求萃取**	90 分钟内摸清一条真实流程，把口头需求写成可验收的问题	**find-skills（按任务找工具）** **library（文档读取与整理）**	企业内部知识库连接器 内部文档系统的只读接口
2	**脏数据工程**	字段对齐、口径固化、质量校验，输出可复算的结果	**sheetagent（表格读写分析）** **pdf / pdfkit-py（抽调表格）**	数据库只读连接器（PostgreSQL / MySQL） ERP / CRM 数据接口
3	**知识资产化（RAG 交付）**	把散落文档变成可召回的知识库，同时守住不出域	**weknora（知识库导入检索）** **library（文档与网盘）**	知识库服务连接器 对象存储 / 网盘连接器
4	**智能体与工作流编排**	把业务规则拆成步骤，封成下次能直接调用的能力	**skill-creator（创建技能）** **expert-manager（专家包）**	多智能体协同 Broker IM 连接器（飞书 / 企业微信等）
5	**系统集成与接口对接**	把客户既有系统接进来，让 AI 自己取数、自己触发	**agent-browser（网页操作抓取）** **playwright-cli（流程测试）**	自建 MCP（stdio / streamableHttp） SaaS 官方连接器
6	**交付物与可复现验证**	输出客户看得懂、敢签字的东西，并且能自己复现一遍	**tencent-pptx（胶片）• tencent-docx（文档）** **sites（发布为可访问链接）**	文档协作连接器（腾讯文档 / 飞书） 邮件连接器

说明：具体可安装的技能与连接器以客户端「技能 / 连接器」页面实际展示为准；自建 MCP 可接入任何实现 MCP 协议的服务，不限于上表。

技能 1 ｜ 现场勘察与需求萃取

现场要做到 90 分钟内摸清一条真实流程，把口头需求写成别人能拿去判断的「可验收问题」

1. **先画地图，别先读内容**
   让它先输出目录树与文件清单，你才知道自己面对的是 40 份还是 4000 份。
2. **只读「入口型」文件**
   挑 3–5 份制度、SOP、报表模板先读，剩下的按需回查，不要一口气全塞进去。
3. **专找说法不一致的地方**
   同一件事在两份文件里口径不同，那条缝就是真实流程的破口。
4. **把所有数字口径抓出来**
   统计周期、含税与否、是否扣退货——这些不写清，后面全是返工。
5. **最后才写成问题清单**
   输出 3 个可验收的问题 + 1 条风险假设，交给客户确认。

WorkBuddy 怎么打

用哪个模式

先「问一问」只读摸底 → 再「想一想」要计划让你确认 → 最后才「做一做」

怎么补上下文

@ 引用文件、拖拽上传、截图 Ctrl+V 粘贴；一次给准，比多给十份有用

用什么技能

find-skills 按任务找工具；library 读文档、建目录、做看板

接什么 MCP

企业知识库连接器、内部文档系统只读接口——让它自己去查，别靠你复制粘贴

验收动作 把问题清单给一个没进过现场的人看，他能判断出「这件事值不值得做、做完怎么算成功」——做不到就说明你还没萃取完。

技能 2 ｜ 脏数据工程

现场要做到 把「字段对不上、口径说不清」的原始数据，变成换个人也能复算出同样数字的结果

常见脏数据	你要求它做的动作	输出物	验收锚点
字段名不统一	先让它出字段映射表，你确认之后再合并	**字段映射表**	源表每个字段都能对上，没有「猜」出来的列
口径不明	统计周期、含税与否、是否扣退货，全部写死	**口径说明**	拿数字能回算出结果，换个人算结果一样
缺失与重复	先报告缺失率与重复条数，再决定怎么处理	**质量报告**	缺失率超过 5% 必须人工确认后才能继续
结果无法复算	抽两条手工算一遍，和它的结果对比	**抽样回算记录**	差异为 0，或者差异原因写在旁边

用哪些 Skill

- sheetagent · 表格读写、公式、透视、图表
- pdf / pdfkit-py · 从 PDF 和扫描件里抽表
- tencent-docx · 把口径与字段说明写成文档

接哪些 MCP

- 数据库只读连接器（PostgreSQL / MySQL）
- ERP / CRM 数据接口
- 共享盘 / 网盘连接器，直接读原始文件

验收动作

- 挑两家门店手工算一遍，对得上才算过
- 把口径写成一句能直接贴进报告的话
- 把清洗过程存成可重跑的脚本或技能

技能 3 ｜ 知识资产化（RAG 交付）

现场要做到 **把散落在网盘、邮箱、个人电脑里的文档，变成客户自己也能维护的知识库**

原始文档
制度、SOP、FAQ、历史问答，先把清单列出来，别急着入库

谁来干：人

→

清洗归一
去水印、清失效链接、统一标题层级，产出可读的 Markdown

谁来干：WorkBuddy

→

分块入库
按语义切块、打标签、补标题，导入知识库并建索引

谁来干：WorkBuddy

→

召回验收
用 20 个真实问题去问，看答得对不对、有没有编

谁来干：人 + WorkBuddy

→

交付使用
写使用说明与更新机制：谁能改、怎么改、多久更新

谁来干：人

用哪些 Skill

- weknora · 知识库导入、检索、召回测试
- library · 文档与网盘读写、建目录做看板
- pdf / pdfkit-py · 从 PDF 与扫描件取内容

接哪些 MCP

- 知识库服务连接器
- 对象存储 / 网盘连接器
- 文档协作连接器（腾讯文档 / 飞书等）

红线与验收

- 文档不许离开客户指定的目录或库
- 召回验收必须用真实问题，不能自己出题自己答
- 无答案时它必须说「找不到」，而不是编一个

技能 4 ｜ 智能体与工作流编排

现场要做到 **把客户嘴里那句「帮我自动跟一下」，拆成五层能被检验的结构，而不是一堆提示词**

L1 • 业务规则 什么算完成、什么绝对不许做、哪些必须人签字 — 人写，人签

L2 • 步骤编排 把规则拆成有明确输入输出的步骤，每步都能单独验证 — WorkBuddy 起草，人确认

L3 • 能力封装 把这套步骤封成一个技能，下次一句话就能调用 — skill-creator

L4 • 触发方式 定时、事件还是手动——先选一种跑起来，别一次做全 — 自动化

L5 • 协同边界 多个智能体分工时，谁负责什么、谁有最终确认权 — 人，写死在流程里

越往上越抽象、越需要人拍板；越往下越具体、越能被自动执行

用哪些 Skill

- skill-creator · 把这次的做法封成技能包
- expert-manager · 把行业经验封成专家，反复用
- find-skills · 先搜有没有现成的，别急着重造

接哪些 MCP

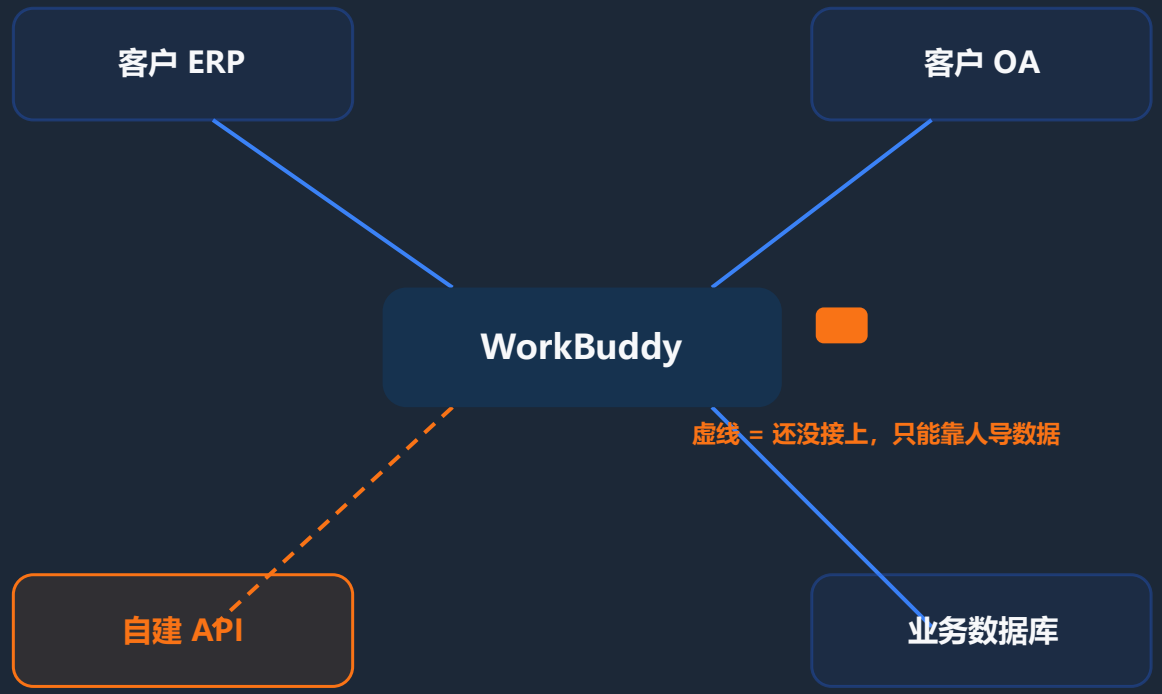
- 多智能体协同 Broker · 跨实例分工与接力
- IM 连接器（飞书等）· 接通知、审批与人工确认
- 业务系统连接器 · 任务的触发源头

编排三条铁律

- 规则由人写，不要让模型替你定义「什么算完成」
- 每一步都要可回退，出错时能停在上一站
- 终审权永远留给人，尤其是对外动作

技能 5 | 系统集成与接口对接

现场要做到 让 AI 自己从客户系统里取数、触发动作，而不是靠你每天手工导出导入



四根线就是四类 MCP 连接器：一个连接器解决一类系统，不要试图用一个连接器通吃。

WorkBuddy 怎么打

两种接法

stdio · 本地进程，适合自建服务；streamableHttp · 远程地址，适合跨机器和团队共用。选哪种只看服务在哪跑。

数据源优先只读

先给只读账号：能查不能改。任何要写回客户系统的动作，单独拉出来走人工确认。

没有现成接口怎么办

按这个顺序退：用浏览器操作界面 → 拿系统导出的文件再读 → 最后才谈改接口。

验收动作

让它在对话里实调一次，拿到真实数据再收工。「配置成功」四个字不算验收。

技能 6 | 交付物与可复现验证

现场要做到 **交出去的东西能让客户自己复现一遍——复现不了，就不算交付，只算演示**

1 汇报件 · 给老板看

只要一页结论 + 三页依据。结论页写清「省了什么、省了多少」，依据页放口径和数据来源。

Skill：tencent-pptx · tencent-docx

2 看板 · 给业务盯

每天更新的那几个数字，加上异常标色。别做花哨仪表盘，做一张能一眼看出今天有没有问题。

Skill：sheetagent

3 可访问链接 · 给客户随时打开

客户点开就能看，不用装环境、不用配依赖。交付当天把链接发出去，比发十个文件管用。

Skill：sites（发布为应用）

4 交接说明 · 给接手的人

谁能改、怎么改、出错怎么办、去哪儿找原始数据。这份东西决定你走了之后它还能活多久。

Skill：library · tencent-docx

复现三件套 ① 数据从哪来、什么口径 ② 怎么跑（一条能复制粘贴的命令或步骤） ③ 结果怎么核对（对哪两个数字）

演练 1 | 90 分钟勘察一个陌生目录

90 分钟

客户丢给你一个共享目录：制度、报表、历史邮件导出，300 多个文件混在一起。你必须在 90 分钟内说清楚「这家公司的月度经营分析到底是怎么出来的」。

输入

一个陌生的项目目录（300+ 文件，格式混杂）

不提供任何解释材料——就像真实的客户现场。

交付物

- 目录树 + 文件分类清单
- 一张业务流程草图（含参与角色）
- 3 个可验收问题 + 1 条风险假设
- 一份「还没搞懂的地方」清单

1 0–15 min · 画地图

让它先输出目录树与文件清单，按类型分组，只看结构不读内容

2 15–40 min · 读入口

挑 5 份最像「流程入口」的文件精读，其余按需回查

3 40–60 min · 找矛盾

让它列出两份文件里同一个口径说法不同的地方

4 60–80 min · 写问题

输出 3 个可验收问题 + 1 条风险假设，别写「优化流程」这种空话

5 80–90 min · 交叉验证

随机抽一份文件，问它「这份在流程里起什么作用」，答不上说明没读懂

验收 把问题清单交给没进过现场的同事，他能判断「这件事值不值得做、做完怎么算成功」。做不到就是没萃取完。

演练 2 | 把脏 Excel 洗成可复算的结果

60 分钟

6 张不同年份的销售表：字段名不一样、日期格式三种、有的含税有的不含、退货有的扣有的没扣。交出一份别人能复算出同样数字的干净数据。

输入

6 张年份不同的销售明细表（Excel），字段与口径各不相同

没有数据字典，没有业务方在场的解释。

交付物

- 字段映射表（6 张表 → 一套字段）
- 清洗后的结构化数据文件
- 口径说明（能直接贴进报告的一段话）
- 质量报告 + 抽样回算记录

1 · 0–10 min · 先要字段清单

让它列出 6 张表的全部字段名和各自两行样例，别急着合并

2 · 10–25 min · 出字段映射表

把 6 张表映射到一套统一字段，你确认之后它才允许继续

3 · 25–40 min · 把口径写死

统计周期、含税与否、退货规则，一条条写进结果说明里

4 · 40–50 min · 出质量报告

缺失率、重复条数、异常值；缺失率超过 5% 必须人工确认

5 · 50–60 min · 抽样回算

抽两行手工算一遍，和它的结果比对，对不上就带着具体数字让它重做

验收 换一个人只看你的口径说明重跑一次，结果完全一致；抽样回算的差异为 0，或者差异原因写在旁边。

演练 3 | 自建一个 MCP 连接器

45 分钟 客户提供了一份内部系统的接口文档。你要让 WorkBuddy 自己能查这个系统的数据，而不是每天靠你手工导出再贴给它。

1. **先确认接口真的能用**
   拿到文档或 curl 一次，确认它返回真实数据再开工，别拿 mock 数据当依据
2. **只暴露 1–3 个动作**
   一次接太多工具，模型会选错。先接「查一条」这一个动作，跑通再加
3. **写进 mcp.json**
   路径固定：~/.workbuddy/mcp.json。远程服务用 HTTP 模式，本机进程用 stdio
4. **回到连接器页点「信任」**
   不点信任，配置写得再对也用不起来——这一步最容易被漏掉
5. **在对话里实调一次**
   让它查一条真实数据给你看。拿到数据才算接上，「配置成功」不算

~/.workbuddy/mcp.json

```
// 方式一 · streamableHttp — 接客户的远程服务
{ "mcpServers": { "customer-erp": {
    "type": "streamableHttp",
    "url": "https://erp.customer.com/mcp",
    "headers": { "x-api-key": "…只读密钥…" } } } }
// 方式二 · stdio — 接跑在本机的自建服务
{ "mcpServers": { "internal-db": {
    "command": "python",
    "args": ["D:/mcp/db_server.py"] } } }
```

安全提醒 配置文件里就是明文凭证：一律用只读账号，别把生产写权限写进去，更别把这份文件放进 Git。

验收 连接器状态显示已连接，并且在对话里实调一次能拿到真实数据；拔掉密钥后它必须报错而不是编一个结果。

演练 4 ｜ 配一条每日巡检自动化

40 分钟 客户希望你每天早上盯一遍关键数据，有问题再叫人。目标不是「每天提醒我」，而是「没事的时候彻底安静」。

1 先定「什么算异常」
阈值必须由人给。让它自己定阈值，你永远说不清为什么报警

2 先手动跑通一次
手动跑一遍确认逻辑对，再挂定时。反过来的返工成本高得多

3 写自包含的描述
自动执行时它看不到你的上下文：路径、口径、阈值、例外都要写进去

4 挂定时 + 选通知方式
每天固定时间跑，只在命中异常时发消息，正常就不要出声

5 自己收到一次才算成
等它真给你推了一条，并且内容能看懂，这条自动化才算交付

自动化 · 任务描述（直接抄这段）

```
每天 08:30 检查 D:/customer/sales/ 下的日销表，规则：
1. 某门店当日销售额 < 近 30 日均值的 60%
2. 当日缺失记录 > 5 条
命中任一条 → 发飞书消息给我，附：店名、当日值、均值
没有命中   → 不要发任何消息
口径：含税销售额，退货已扣减；均值按最近 30 个自然日
若目录里没有当天文件，直接报「数据未到」而不是跳过
```

常见误配 把「每天提醒我一下」配成自动化——那叫打扰，不叫值班。通知条件必须写成可判定的规则。

验收 你自己真实收到过一条告警，内容包含店名、当日值与均值；正常一天里它一条消息都不该发。

交付前的七项自检

第一项和第四项不过，其他都别谈 这两项决定了客户是在「用」还是在「看你演示」。

1 可复现
换个人按你的说明走一遍，能跑出同样的结果

2 口径写明
数字怎么算、以哪一天为界，写在交付物里而不是你脑子里

3 权限最小
它拿到的是刚好够用的权限，不是管理员账号

4 有验收动作
客户自己能验一遍，不用听你的口头保证

5 有回退方案
出问题怎么退、退到哪一步、退完谁接手

6 会解释
每个结论都能说出依据是哪份文件、哪一条数据

7 有交接说明
你走了以后，谁改、怎么改、出问题找谁

交付前先问自己一句

如果我现在离职，这套东西还能活三个月吗？答案是否，就去补第 4 项和第 7 项。

第 3 章 · P23–P31 · 60 分钟

工具箱
Skill 与 MCP 全清单

技能管「怎么做」，连接器管「连到哪」。

搞清楚这两件事的分工，你就知道自己缺的是哪一种。

本章产出 属于你自己的工具箱清单：装什么、写什么配置、什么必须关掉



Skill：装、开、关、卸四件事

技能是给 WorkBuddy 加「具体干活的手」，不是知识 它封装了一组脚本与工作流，调用时以你自己的身份执行。



界面为示意，实际以客户端显示为准

装 · 三条路

- 上传技能：导入本地技能包，拖进去即可，系统自动完成配置
- 查找技能：描述你的任务，它去找有没有现成的
- 创建技能：描述任务，让它直接把技能造出来

开关 · 关 ≠ 卸

- 关闭只是暂停模型自动调用，技能文件与配置都保留
- 开关状态记在你账号里，多端同步；不动 SKILL.md 原文件
- 建议只开当前任务需要的，减少误调用

卸载 · 真的删

- 彻底删除技能本体与开关记录，支持批量操作
- 卸载前确认没有别的任务还在依赖它
- 拿不准就先关掉跑一周，别急着卸

安全 · 官方优先

- 优先使用官方推荐技能
- 第三方技能先看权限申请、来源说明和脚本内容
- 涉及删文件 / 批量写入 / 改系统配置，先小范围验证

FDE 常用的 12 个 Skill

12 **按「现场什么时候用」分四类，不按功能分类**

现场缺的不是功能清单，是「现在这个场景该用哪个」。

装之前先想：这个任务真的需要它吗

A · 文档产出

tencent-pptx

客户汇报、方案评审要交胶片时；有母版就照母版出

tencent-docx

交付说明、运维手册、口径文档要正式成文时

pdf / pdfkit-py

客户给的合同、报表、扫描件要抽正文和表格时

B · 数据处理

sheetagent

客户丢来一堆表要合并、透视、出图、算指标时

askdata

客户问「上个月哪个渠道涨得最快」，要直接出数

ml-models

要做预测、打分、排序这类表格建模任务时

C · 网页与集成

agent-browser

系统只有界面没有接口，要登录、翻页、下载时

sites

交付物要变成客户点开就能看的链接时

library

要建目录、建数据表、做看板、管网盘文件时

D · 领域与自建

weknora

交付知识库：导入文档、跑召回测试、验答案

find-skills / skill-creator

先搜有没有现成的；没有就当场造一个

expert-manager

把这次沉淀的方法论封成专家包，团队复用

这是常见组合，不是完整清单：实际以客户端技能市场、以及你团队已沉淀的自建技能为准。D 类里的每一项，都可能是你自己造出来的。

MCP 连接器：四步接上外部系统

连接器就是 WorkBuddy 的「插座」 凡是实现了 MCP 协议的服务都能接进来——客户内部 API、第三方 SaaS、你自己写的工具，都算。

1 打开连接器页

桌面端左侧导航「连接器」（插头图标），不是「技能」页

2 找到你要的服务

按分类浏览或搜索。官方连接器优先，能覆盖的别自己造

3 完成认证

OAuth 扫码 / API Key / 账号密码，按提示走；凭证由你提供

4 点「信任」激活

装完不会自动启用，必须点信任；之后在对话里就能直接调用



界面为示意：连接器分类与名称以客户端实际展示为准

配置都写在这一个文件里

```
// 全局配置文件：~/.workbuddy/mcp.json
{
  "mcpServers": {
    "your-service": { "command": "…", "args": ["…"] }
  }
}
```

最容易漏的一步 装完必须点「信任」才会启用。状态显示异常时，先查认证凭证有没有过期，再看配置格式对不对。

FDE 常用的三类 MCP 连接器

第一类能直接用，第二类决定你能拿到什么数据，第三类决定你能不能进客户现场

官方 SaaS 连接器

装完即用，认证走 OAuth

- 文档协作（腾讯文档 / 飞书等）
- 即时通讯与审批
- 邮件
- 日历与会议

把交付物直接送进客户已有的工作流，而不是发个文件让人自己传

数据源连接器

FDE 的主力，因为数据都在这儿

- 数据库：PostgreSQL / MySQL（务必只读）
- 数据仓库与 BI
- 对象存储 / 网盘
- ERP / CRM 数据接口

让它自己去取数，替掉你每天的手工导出导入

自建私有连接器

客户内部系统，官方一定没有

- 客户内部 API（stdio / streamableHttp）
- 只有界面的老系统（用浏览器兜底）
- 你自己写的工具服务
- 跨实例协同（多智能体 Broker）

接完必须在对话里实调一次，拿到真实数据才算接上

经验值 连接器越少越好用：一个任务最多接 2–3 个。接太多，模型会在不该调的时候调错工具。具体可用清单以客户端连接器页为准。

自己写 MCP：字段逐条对照

配置写错，九成是这六个字段里的一个 配置文件只有一个 `~/.workbuddy/mcp.json`

字段	写什么	常见错法
`mcpServers`	顶层容器，所有服务都挂在它下面	名字拼错，写成 mcpServer
服务名（key）	你自己起的标识，对话里靠它区分不同系统	用中文、空格，或者起了重名
`command + args`	stdio 模式：要跑的命令，加上脚本的绝对路径	写相对路径，换个目录就失效
`env`	传给这个进程的环境变量：账号、令牌、地址	把生产写权限直接写了进去
`type + url`	HTTP 模式改用这两个：streamableHttp + 服务地址	和 command / args 混着写
`headers`	HTTP 模式的认证头，通常放 API Key	密钥进了 Git 仓库或截图

~/.workbuddy/mcp.json

```
{
  "mcpServers": {
    "customer-crm": {
      "type": "streamableHttp",
      "url": "https://crm.customer.com/mcp",
      "headers": { "x-api-key": "…只读密钥…" }
    }
  }
}
```

① **连不上** 先确认地址通不通（浏览器或命令行访问一下），再看 command 指的文件在不在

② **连上但调不动** 查两件事：密钥是不是过期了；这个服务有没有在连接器页点过「信任」

③ **调了但结果不对** 用最小参数手工调一次，先排除是模型传参的问题，还是接口本身返回就有问题

连接器安全：三层，一层都不能省

连接器是唯一能让 AI「碰得到东西」的入口，所以安全也都在这里

01
信任层

装完必须点「信任」才会启用。信任的含义是：你明确知道它能碰到什么、碰不到什么。

检查动作 装完问自己一句：这个连接器能读到哪里、能写到哪里？答不上就先别用。

02
权限层

只给这个任务需要的权限。只读就别给写，能限定目录就别给全盘，生产环境一律先只读。

检查动作 动手前写下「这个任务最少需要什么」，多出来的一律不给。

03
凭证层

密钥不进 Git、不进对话、不写进截图。用只读账号，交付前轮换一次。

检查动作 交付时把密钥从配置文件里摘出来，另行当面交接，不跟着文档一起发。

安全层数

3

任何一层跳过
都等于没做

只点信任不管权限，等于把钥匙插在门上；只给最小权限不点信任，它根本用不起来；凭证乱放，前两层再严也白搭。

一句话：能只读就不给写，能限定就不放开，能少接一个就少接一个。

提醒：连接器与技能都以你的身份执行外部动作。涉及资金、外发消息、删除数据三类操作，永远保留人工确认点。

Skill / MCP / 专家 / 自动化，我该用哪个

别问「哪个更强」，问「我现在缺的是哪一样」 四者管的事完全不同，选错方向比选错工具贵得多。

我缺的是「干这个活的手」
→ **装 / 造一个 Skill** 脚本、流程、动作类能力

我缺的是「够不着那个系统」
→ **接一个 MCP 连接器** 数据源、客户系统、第三方服务

我缺的是「这件事该怎么做才专业」
→ **换一个专家** 方法论、判断标准、行业经验

我缺的是「这件事不该我每次记着」
→ **配一条自动化** 定时、事件触发、持续巡检

四种典型选错

× **把 MCP 当 Skill**
写了一大堆连接配置，其实只差把动作封装成一个技能

× **把专家当权限**
以为装了专家就能访问数据，实际什么都没接上，它只是换个说法

× **把自动化当提醒**
「每天提醒我一下」——产出的是每天一条噪音，不是一条判断

× **把 Skill 当知识**
指望技能回答业务问题，实际它只会执行动作，不会替你做判断

记住一条：专家只改变「怎么想」，连接器才改变「能碰到什么」。别拿专家去补权限的坑。

找不到趁手的工具，走这三条路

1·2·3　顺序不能反：先搜、再造、最后才接

90% 的「缺工具」，第一条就解决了；直接跳到第三条，是最常见的时间黑洞。

路径一

技能市场搜

先看有没有人已经做过这件事。输入你的任务描述，让它去找现成的技能。

成本　几分钟

代价　不一定贴合你的场景，可能要装几个试试

路径二

让它按需求创建技能

把这次一步步做出来的流程描述清楚，让它直接造一个技能包出来。

成本　一次对话

代价　生成的技能必须自己验收，别直接给客户用

路径三

自己写 MCP 连接器

只有当你缺的是「够不着某个系统」时才走这条，而不是缺一个动作。

成本　半天到一天

代价　要长期维护，还要管凭证与轮换

判断小技巧　能用一句话说清「要它做的动作」，就走路径一或二；说不清、只能描述「要看哪个系统的什么数据」，才是路径三。

第 4 章 · P32–P38 · 120 分钟

实战演练营
做出能交出去的活



四张任务卡，分组完成。每张卡都写死了输入、交付物和验收标准——

验收标准就是评分线，卡上怎么写，就怎么评。

本章产出 4 份可演示的交付物 + 一份五维评审表

四张任务卡，按小组认领

A 和 B 偏业务，C 和 D 偏工程；小组按自己的现场情况挑一张 难度 ★ 到 ★★★★★，卡上标的就是上限，不要求全做完。

A 客户汇报包 业务向 ★★☆☆☆

输入
会议记录 + 一份原始数据

交付物
一页结论 + 一份汇报胶片

主力工具：tencent-pptx • sheetagent

B 数据接入与问数 数据向 ★★★☆☆

输入
一份 CSV + 一个业务问题

交付物
可复现的查询链路 + 口径说明

主力工具：sheetagent • askdata

C 知识库与智能体 AI 向 ★★★★

输入
30 份业务文档

交付物
可召回知识库 + 会拒答的问答入口

主力工具：weknora • library

D 集成与自动化 工程向 ★★★★★

输入
一个 HTTP 接口

交付物
MCP 连接器 + 每日自动化

主力工具：MCP（stdio / HTTP）• 自动化

任务卡 A ｜ 客户汇报包

A **业务向 · ★★ · 45 分钟** 你旁听了一场客户月度经营会，手上有会议记录和一份原始销售明细。老板要一份「能直接上会」的材料。

输入

· 会议记录（文字，含两处口径不一致）

· 一份原始销售明细（未清洗）

交付物

- 一页结论页：省了什么、省了多少、下一步
- 6–8 页汇报胶片，可直接投屏
- 一页口径与数据来源说明
- 一份「不确定的地方」清单

边界

不许把原始明细整张贴进胶片；每个数字都要能追到来源。

1 **读会议记录**

抽出老板真正关心的那几个问题，不是所有议题

→

2 **读原始数据**

对齐字段与口径，把不确定的地方单独标出来

→

3 **出结论页**

一个结论 + 三条依据，先给客户口头确认再展开

→

4 **出胶片**

照结论页展开成 6–8 页，母版统一、页码齐全

关键在于第 3 步：先出结论页给客户确认，再展开做胶片

顺序反了，你要返工的就是那 6–8 页；顺序对了，客户确认过的结论页直接变成胶片第 1 页，剩下的只是展开。这是 FDE 做汇报件最省时间的一个习惯。

另外记一条：会议记录里两处口径不一致，别自己选一个——把两处都写进「不确定的地方」清单，让客户拍板。

主力工具：tencent-pptx（胶片）· sheetagent（数据）· 提问模式（读会议记录）

验收标准 老板只看第一页就能说出「省了多少」；随机点开任意一页，你能说出这个数字来自哪张表、哪一列。

任务卡 B ｜ 数据接入与问数

B **数据向 · ★★★ · 60 分钟** 客户问「上个季度哪个渠道增长最快、为什么」。你手上只有一份渠道销售 CSV，没有数据字典。

输入

· 渠道销售明细 CSV（字段名是缩写）

· 三种日期格式混在一起，无数据字典

交付物

- 字段映射表（缩写 → 业务含义）
- 口径说明（周期 / 含税 / 退货）
- 一条可复现的查询链路
- 一张结果图 + 一句话结论

边界

口径不确定就列出来让人拍板，不要自己替客户选一个。

原字段名	业务含义	告诉客户这句话
chan_cd	**渠道编码**	与渠道主数据表的 channel_id 对应，可能有一对多
gm_amt	**含税销售额**	原文含税；不含税需按 6% 反算，口径要写清
dt	**销售日期**	三种格式混用：20260901 / 2026-09 / 2026/9/1

① 要字段清单
先列出全部字段 + 样例值

② 出映射表
缩写还原业务含义，你确认

③ 口径写死
周期、含税、退货规则

④ 出数与回算
抽一个渠道手工算一遍

⑤ 出图与结论
一句话结论 + 支撑数据

主力工具：sheetagent（表格读写与出图）· askdata（自然语言问数转 SQL）

有数据库时才走 askdata：先只读账号接上数据源连接器，再让它把你那句业务问题翻成 SQL。注意第二列「业务含义」必须你确认——它猜错的字段，后面整条链路都是错的。

验收标准 换个人拿你的链路重跑，得到同样的图；结论里每个数字都能对应到映射表里的某个字段。

任务卡 C ｜ 知识库与智能体

C **AI 向 · ★★★★ · 75 分钟** 客户给你 30 份制度、SOP 和历史问答，想做一个「新人敢问、老人敢用」的问答入口。关键不是答得多，而是不许编。

输入

· 30 份业务文档（PDF / Word / 表格混杂）

· 含一份已失效的历史版本，需要你自己发现

交付物

- 一个可召回的知识库
- 一个能回答的问答入口
- 20 题召回验收记录（含答错的）
- 使用说明与更新机制

边界（写在卡上，不靠自觉）

文档不许离开客户指定目录；不许用未确认来源回答；答不出来必须说找不到。

文档清单 先列出 30 份文件的类型与版本，别直接入库 → **清洗归一** 去水印、清失效链接、统一标题层级 WorkBuddy → **分块入库** 按语义切块、补标题、打标签，建索引 WorkBuddy → **20 题验收** 用真实问题去问，记录答对与答错 人 + WorkBuddy → **交付与说明** 写使用说明与更新机制，交出去 人

20 题召回验收记录（节选）	判定
制度里规定几天内报销?	✓ 答对 · 能引到具体条款
去年的差旅标准是多少?	✗ 答错 · 过期版本，需标注生效日期
公司团建预算是多少?	✓ 拒答 · 库里确实没有，说「找不到」

验收标准 20 题里答对 ≥ 16 题；问一个库里没有的问题，它必须说「找不到」而不是编一个答案给你。

任务卡 D ｜ 集成与自动化

D **工程向 · ★★★★★ · 90 分钟** 客户给了一个内部系统的 HTTP 接口。希望你每天早上自动核对一批数据，有异常才通知，没问题就别打扰。

输入

· 一个内部系统的 HTTP 接口文档

· 一个只读密钥（别无其他）

交付物

- 一个可用的 MCP 连接器（对话里能实调）
- 一条每日自动化（含异常判定规则）
- 一次真实的告警记录
- 一份凭证交接说明

1. **确认接口能返回真实数据**
   拿文档或实调一次，确认不是空壳接口再动手
2. **只暴露一个查询动作**
   先接「查一条」这一个工具，跑通之后再考虑加
3. **写 mcp.json 并点「信任」**
   配置写完必须回连接器页点信任，否则不会启用
4. **在对话里实调一次**
   让它查一条真实数据给你看，拿到数据才算接上
5. **配一条每日自动化**
   写清异常判定规则，命中才发通知，正常保持安静

配置 + 自动化描述（可直接抄）

```
// ~/.workbuddy/mcp.json
{ "mcpServers": { "customer-ops": {
    "type": "streamableHttp",
    "url": "https://ops.customer.com/mcp",
    "headers": { "x-api-key": "…只读密钥…" } } } }
// 自动化任务描述
每天 08:30 调用 customer-ops 查前一日工单数;
若较近 30 日均值偏差 > 30% → 发消息给我，附数值;

若目录里没有当日数据 → 报「数据未到」，不要跳过。
```

边界（这张卡最容易出事）

只用只读密钥；任何要写回客户系统的动作，必须保留人工确认点，不许全自动发出去。

主力工具：自定义 MCP（stdio / streamableHttp）· 自动化 · agent-browser（无接口时兜底）

如果接口是外网不可达的，先让客户在能连通的机器上跑一个 MCP server，再把地址填进配置。

验收标准 你真实收到过一次告警；连接器状态为已连接；把密钥摘掉后它必须报错，而不是编一个结果。

成果评审：五个维度，各 1–5 分

评分不看技术含量，只看「交出去以后能不能活」 小组互评 + 讲师复核，每组 10 分钟答辩。

维度	这一项在问什么	1 分 · 不合格	5 分 · 可交付
能用	客户现在能不能拿它干活，而不是看你演示	只能演示，你一停手就没了	客户自己独立跑通过至少一次
可复现	换个人能不能得到同样的结果	只有你本机能跑，口径说不清	说明写完，别人照着跑结果一致
说人话	交付物客户能不能看懂、敢签字	满屏术语和内部黑话	第一页就能说出结论和数字
边界清楚	能做什么、不能做什么写没写清	全靠口头交接，出事没人认	边界写在交付物里，客户确认过
好交接	你走了以后它还能不能活着	只有你知道怎么改	写清改哪里、出问题找谁、怎么退

18
合格线

五维合计 ≥ 18 分算交付合格；任意一项打 1 分，整体直接判不合格——因为那一项通常正是客户三个月后会来找你的原因。

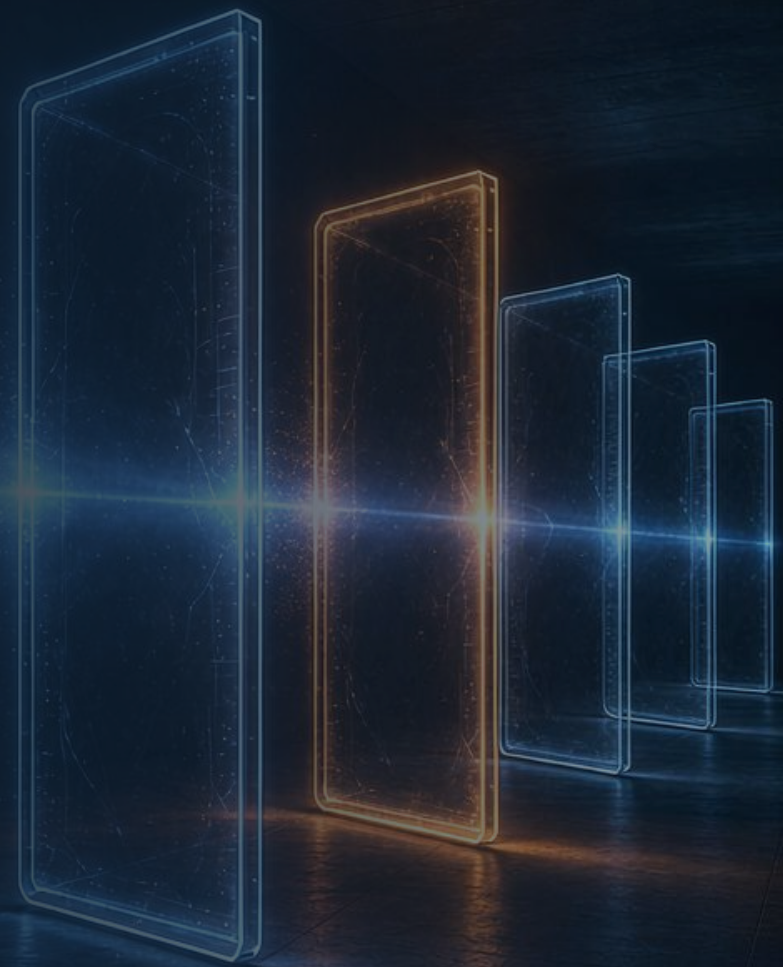
第 5 章 · P39–P44 · 30 分钟

避坑、边界与 90 天路线

前面的错都还能补，这几条补起来很贵：

安全边界、权限配置、以及「你走了之后」的交接。

本章产出 属于你自己的 90 天能力路线图与四道安全闸门检查动作



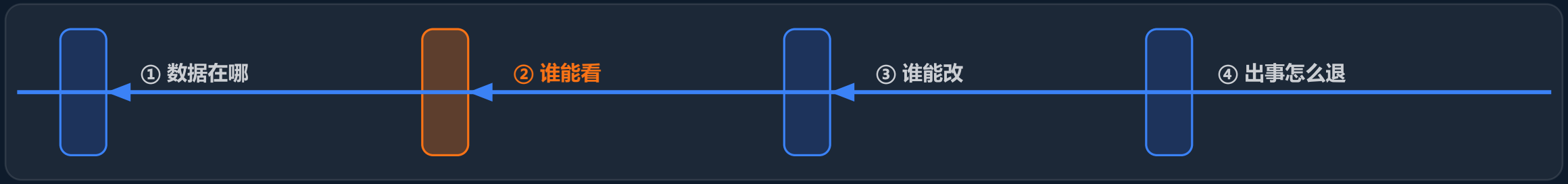
七个高频坑，每个都有解

前四个坑能补，后三个补起来很贵 尤其是「数据出域」和「没有回退」，出一次就可能丢掉一个客户。

#	坑	后果	怎么避开
1	**把演示当交付**	屏幕上一跑通就以为完事，客户手上其实什么都没有	**交付前让客户自己独立跑一遍，跑通才算完**
2	**权限给太满**	图省事直接给管理员，出事时说不清是谁改的	**只给这个任务需要的权限，多的一律不给**
3	**口径没写**	数字对不上时无人可辩，返工只能从数据重来	**口径写进交付物，不留在你脑子里**
4	**数据出域**	把客户数据贴到不该去的地方——这类错不可逆	**先用脱敏样本跑通，正式数据只在客户环境内处理**
5	**没有回退**	上线即不可退，出问题只能现场硬修，风险全压在你身上	**写一个回退步骤，最好自己演练过一次**
6	**不写交接**	只有你知道怎么改，你一休假它就停摆	**写清改哪里、找谁、怎么改，交付时一起给**
7	**一个人扛**	需求、方案、实现全自己来，卡住了没人能接	**关键判断找人复核一次，别自己兼任裁判**

自检方式：交付前一天，把七条挨个念一遍，任何一条答不上来，就别交。

四道闸门：交付前必须全部过一遍



① 数据在哪

这份数据现在存在哪里？允许存在哪里？

检查动作

列一张数据落地清单：本地 / 客户内网 / 云端，每一项写明允许范围。

✕ 把客户明细存进自己的云盘方便随时调

② 谁能看

谁会看到处理过程和结果？

检查动作

检查交付物里有没有超出授权范围的内容——截图和分享链接尤其要看。

✕ 截图里带着客户客户名单，直接就发群里了

③ 谁能改

谁有权限改这套东西？改之前要不要有人批？

检查动作

写清权限清单，以及变更需要谁确认；生产环境一律先只读。

✕ 为了方便调试，给了一个永久的管理员密钥

④ 出事怎么退

出问题退到哪一步？谁执行？

检查动作

写一个回退步骤，并且自己演练过一次——没演练过的回退方案等于没有。

✕ 「万一有问题就手动改回来」，谁来改不写

顺序不能跳 数据在哪决定你能不能开始；谁能看决定你能交付什么；谁能改决定上线方式；怎么退决定你敢不敢上线。

从会用到能交付：四级台阶

大多数人卡在第一级和第二级之间，而且不自知 判断自己在哪一级，只看一个标准：别人能不能接手你交出去的东西。

会用
能把一件活交代清楚，并拿到可用结果

会拆
能把模糊需求拆成可验收的步骤

会造
能造出可复用的 Skill / MCP / 自动化

会讲
能把成果讲到客户敢签字

1 会用的判据

不看教程也能独立完成一次端到端任务

常见停滞点

停在「会聊天」：一遇到交付就退回手工做

2 会拆的判据

你写的问题清单，别人能拿去判断值不值得做

常见停滞点

直接开做，做到一半才发现需求没问清

3 会造的判据

同一件事第二次做，用时不超过第一次一半

常见停滞点

每次都从头来，成果不沉淀、不封装

4 会讲的判据

客户只看你的一页结论就能做决定

常见停滞点

技术很好但讲不清，交付总卡在验收环节

往上走的唯一办法：不是学更多功能，而是每做完一件事就多沉淀一点可复用的东西。

30 / 60 / 90 天：怎么真的练出来

每个阶段只求一件事落地，不求全面 共同要求：每个阶段都必须留下「别人能接手」的东西，否则等于没做。

30 天 把六项技能各跑一遍

要做的事

- 每周挑一项技能，找一个真实的小事练一遍
- 从演练 1 到演练 4 依次走，笔记只记「卡在哪」
- 每周留半天，把这一周的做法写成一页

验收标准

六份演练交付物留在你的样例库里，随时能拿出来给人看

该沉淀什么资产

一份自己的「常用指令与配置」清单

60 天 独立接一个小任务

要做的事

- 从勘察到交付完整走一遍，全程按七项自检
- 第 3 周主动请一个同事挑毛病，把问题写进下一版
- 交付前把口径、边界、回退写成三页说明

验收标准

客户独立跑通过一次；你收到过一条真实告警或一份书面确认

该沉淀什么资产

一套可复用的交付模板：结论页 + 交接说明

90 天 沉淀一套自己的工具箱

要做的事

- 挑你最常重复的三件事，各封成一个 Skill 或 MCP
- 把最常被问的问题整理成一份问答，接上知识库
- 把这三个月的方法论写下来，团队内讲一次

验收标准

这三件事第二次做，用时不超过第一次的一半

该沉淀什么资产

一份可分享的方法论（建议封成专家包）

提醒 别把三个月都花在「学更多功能」上。FDE 的成长曲线来自真实交付的次数，不是功能清单的长度。

收尾 · 一句话带走

1

「客户不会为你的工具买单，只会为你替他们省下的那件事买单。」

所以从明天开始，只做一件事：找一件你每周都要重复的活，用今天学的六项技能把它跑通一遍， 然后把做法沉淀下来。做满三次，你就不再是「会用工具的人」了。

六项技能 → 装 / 造 / 接 / 沉淀

七项自检 → 交付前一天念一遍

四道闸门 → 数据 / 可见 / 可改 / 可退