AI 应用能力作品集

把 AI 能力拆成
可交付、可验收的产品规则

AI Builder-产品-2027秋 · 应聘材料 · 曾庆烨

学校专业	华南师范大学 · 经济与管理学院 · 信息管理与信息系统（本科 · 2027 届）
求职方向	AI 产品经理实习 / 校招 · AI 应用与知识工作流方向
个人网站	myricardo.online
GitHub	github.com/Ricardo-ye
本册正文三篇	企业级 RAG 评估工作台 · RecNote Agent · 暖心护家庭协作小程序 附录索引表另列两项同等可核验：FDE Atlas 企业 AI 案例图谱 · Z1Space（团队项目）
材料版本	2026-10-22 · 提交版（已按公开仓库与站点逐项核对修订）

这份作品集想证明一件事：AI 产品的核心交付物不是「模型效果」，而是**可被检查的产品规则**——先定义问题与验收条件，再把方案落成可运行的交互、服务与测试链路，最后留下可追溯的证据。

三个项目分别对应这条链路的三段：**RAG 评估工作台**解决「效果怎么被量化」；**RecNote Agent** 解决「Agent 的不确定性怎么被管住」；**暖心护** 解决「一个 0→1 产品怎么被收敛并验收」。每个项目我都写明了**不适用的结论**——这是我判断自己交付物是否可信的方式。

评审者从哪里开始（四个入口，都不需要装环境、不需要自备 Key）：
① **开始体验**　myricardo.online —— 三个项目的案例拆解、界面设计稿与诚实边界，静态站直接打开
② **看实跑**　本册第二条作品：2 分 55 秒带字幕演示视频，九段均为真实运行录屏（含评测跑出 0 分并主动报警的画面）
③ **查证据**　github.com/Ricardo-ye —— 本次申请的作品索引置顶在个人主页首屏，每个指标都带报告文件与复现命令
④ **看设计系统**　github.com/Ricardo-ye/fde-atlas（已开 GitHub Pages）—— 61 项单测 + 59 项浏览器集成测试，内含可独立复用的 UI 设计系统与可视化 styleguide

RAG 评测体系　Agent 工作流　人在环治理　可观测性　0→1 产品收敛　验收标准定义

CONTENTS

目录与岗位对应关系

先给结论：正文三篇分别打 JD 的哪一条、各用什么证据支撑；附录索引表另列两项同等可核验的支撑项目。

JD 职责	对应项目	支撑证据（可当场复核）
1. 识别适合 AI 重构的场景 判断用户价值 / 商业价值 / 技术可行性	企业级 RAG 评估工作台 第一篇	企业问答场景拆解为 5 条路由（拒答/闲聊/任务流/知识库/开放域）；把「技术可行」与「效果可信」分成两套口径分别验收
2. 把大模型/Agent/工作流转为可落地方案 定位与功能规划	RecNote Agent 第二篇	五步 Agent 管线 + 受限工具集（模型不能直接写正式笔记）；7 个页面、60+ 项无头浏览器断言全部通过
3. 0→1 构建与迭代 原型验证与迭代优化	暖心护家庭协作小程序 第三篇	7 场景 → 15 页原型 → 4 模块收敛；**50/50 自动化测试通过** + 6 视口渲染验收；F01–F06 六个缺陷修复并有回归覆盖；明确写出未完成的平台验证项
4. 建立可量化的评估标准 价值评估与复盘	企业级 RAG 评估工作台 第一篇（核心）	12 题固定评测集；确定性指标与 LLM 判官指标分口径呈现；本轮修复 9 个问题，其中 2 个是「看起来成功、实际是错的」静默错误
5. 可复用能力沉淀 支撑规模化复制	三个项目共用同一套方法	「数据/报告为唯一事实来源」「文档数字必须可复现」「先质疑度量再谈结论」三条规矩贯穿三个项目

为什么这样组合

本次对标岗位的 JD 中，「可量化的 AI 产品评估标准」是最稀缺、也最难临时抱佛脚的一条。多数候选人能演示 AI 能力，但很少能拿出「我定义过指标口径、并发现过指标本身有问题」的证据。所以第一篇给足篇幅（4 页），后两篇各 3 页，前两篇之间不放「其他小项目」以免稀释。

上表左列按产品经理通用职责项组织，可直接对照任意 AI 产品经理 JD；若目标岗位的职责表述不同，以 github.com/Ricardo-ye 个人主页首屏的「本次申请作品索引」为准。

01　企业级 RAG 能力评估工作台 · 可量化的评估标准从0 到 1　P3 – 6

02　RecNote Agent · 把 Agent 的不确定性写成产品规则　P6 – 8

03　暖心护 · 家庭协作小程序的 0→1 收敛与验收　P8 – 10

04　附：其他作品索引与工程化规范　P10

关于这份材料的三个使用建议：

- **先看第 2 页的对应关系表。**面试时可以直接按表里的顺序讲，不必等对方翻到对应章节。
- **每篇的「诚实边界」不要跳过。**这些限制声明本身就是我判断交付物可信度的方式，也是简历里没篇幅展开的部分。
- **所有数字都可现场复现。**RAG 章节的每个指标都标注了来源报告文件与复现命令；RecNote 与暖心护的断言数量、测试通过情况都有本地记录。

项目 01 • 评估体系

企业级 RAG 能力评估工作台

我的角色：独立项目 · 产品设计与验收设计（AI 辅助实现） | **周期：**2026-09 至今，持续迭代 | **技术栈：**FastAPI · BM25 + FAISS + TF-IDF/SVD 混合检索 · DeepSeek-chat / reasoner · Streamlit · pytest

> **对应 JD：**职责 1「识别适合通过 AI 重构的场景」+ 职责 4「建立可量化的 AI 产品评估标准」。这是本册权重最高的一篇。

> **读这一篇前，先分清两个仓库，避免互相背书：**本篇涉及两个独立项目——**intelligent-cockpit-rag**（座舱评测：Streamlit + rag_eval 引擎，负责指标口径与真实模型实验）与 **enterprise-rag-workbench**（FastAPI + SQLite 内置界面，负责知识治理与交互，默认 hashing-v1 离线 mock 演示）。**本篇的界面截图来自后者，界面右上角标注 local • mock；本篇的指标数字来自前者。**后者的 mock 结果不能给前者的真实模型实验背书，前者的真实模型结果也不能证明前者已完成真实模型验收——这两句话必须一起说，才成立。

问题：企业知识问答最容易骗人的地方，是「有数字」不等于「有效果」

企业问答场景看似适合 RAG 重构——人工客服重复回答率高、知识分散、更新频繁。但真正开始做之后会发现，评估本身就是最大的坑：一个看起来 90% 的检索命中率，可能掩盖了「答案全是空的」；一个 faithfulness=1.0 的指标，可能只是因为抽取式答案本就抄自原文，属于「自证清白」。

所以我把这个项目的定位不是「做一个问答系统」，而是**「做一套能证明问答系统是否真的有效的评估工作台」**——先质疑度量，再谈结论。

产品方案：把「一次问答」拆成可验收的五段链路

① 知识摄入	Markdown / UTF-8 文本 / PDF 解析，标题感知切片；SHA-256 幂等入库 + 跨源去重；支持版本、软删除、FAQ 与知识图谱三元组
② 分层路由	安全检查 → 消费/业务分类 → 领域与子意图识别 → 路由到 拒答 / 闲聊 / 任务流 / 知识库问答 / 开放域 RAG
③ 多通道检索	语义 / 词法 / 网页 / 知识图谱 / FAQ 并行检索 → RRF 融合 → 去重 → 可插拔重排 → 带编号的证据
④ 受控生成	证据约束提示词、引用与事实校验、合规审计、低置信拒答、一次自动修复尝试、人工兜底
⑤ 可观测底座	就绪探针与备份、API Key 保护、基础限流、可信主机、Prometheus 指标、SLO、答案质量记录

评估标准：两套口径必须分开呈现

这是整个项目的核心设计。确定性指标（hit_rate、precision@k、mrr、ndcg、token_f1 等）用于回归对比；LLM 判官指标（faithfulness 拆原子论断逐条核查、answer_relevancy 反推问题式判定）用于对外汇报。两者混在一起是最常见的评估事故。

指标	离线链路	全 LLM 链路	口径说明
hit_rate	0.9167	1.0000	确定性 · 12 题固定评测集
precision_at_k	0.1833	0.5000	确定性 · LLM 重排更准
mrr / ndcg	0.7778 / 0.8135	1.0000 / 1.0000	确定性
token_f1	0.3619	0.7275	确定性 · **+101%**
rouge_l	0.3819	0.7447	确定性 · **+95%**
faithfulness	1.0000 词法代理	0.9722 LLM 判官	**口径升级为语义判定**——离线的 1.0 只是「自证清白」
answer_relevancy	0.1215 词法代理	0.3146 LLM 判官	数值偏低属正常：信息含量不足以唯一反推原问题

> **必须声明的归因边界：**上表两列**不可逐行对比。**离线链路用默认配置（rerank=none、compress=off、词法代理），全 LLM 链路是rerank=llm、compress=on、LLM 判官。它反映的是两条链路的端到端差异，**不是单变量消融结果。**要严谨归因需固定其余变量再扫描。我在文档里也专门写明了这一点，避免被复核者用一条命令打假。

发现的问题：2 个「看起来成功、实际是错的」静默错误

本轮核查 8 个维度、修复 9 个问题、最终回归 7/7 通过、真实 LLM 全链路 12/12 成功（96 次真实调用，0 重试 / 0 截断 / 0 异常）。其中两个问题我认为最能体现产品判断力：

问题	现象与根因	我的修复与产品判断
P0-1　空答案被计为成功	思维链模型（deepseek-reasoner）在 max_tokens 较小时返回 `finish_reason="length"` 且 `content=""`。旧代码把空字符串原样写入报告，**指标照常计算、题目标记成功**——数字全在，但答案全是空的。根因是推理 token 与正式输出共享同一预算，上限 32 时推理阶段就耗尽预算。	新增 `MIN_MAX_TOKENS_REASONING = 2048` 与模型识别；截断或空内容一律**抛错而非降级为成功。**产品上判断：**宁可让一次运行失败，也不能让一个假的成功进入汇报材料。**
P1-3　耗时口径失真（虚高 5.3 倍）	上下文压缩在 generate() 内部执行，但评估用墙钟时间包裹整个调用，导致「平均生成耗时」混入了压缩耗时。实测：旧口径 5674.3 ms中压缩占 4598.0 ms（81%），真正纯生成只有 1076.3 ms——**生成延迟被报高了约 5.3 倍。**	压缩与生成分别计时，CLI 与 UI 三处同步展示拆分口径；启用压缩时显式提示「平均生成含压缩」。产品上判断：**指标口径错了，指标越漂亮越有害。**
P2-4　文档数字腐化	三份文档的指标表沿用旧基线，且同一份报告内第 56 行写 0.4220、第 168 行写 0.421971——**自相矛盾**，说明数字是从不同批次手工粘贴的。	全部按实测值回填 + 标注报告文件与复现命令；把 data/reports/*.json 定为唯一事实来源。反思：**文档里的数字若不是由脚本从报告自动生成，就一定会腐化。**

项目 01 • 续

RAG 评估工作台 • 验证与可复现

验证方式：静态审查 + 真实联网调用 + 端到端实跑

核查全程**不使用任何模拟数据或占位值。**网络连通性采用分层验证：DNS → TCP → TLS → 裸 HTTP（绕过 SDK 掩蔽）→ SDK → 业务调用，避免只测 SDK 而漏掉网络层问题。

验证层	结果
网络可达性	DNS 解析 → TCP 443 → TLSv1.3 → 裸 HTTP 200（0.93s），逐层留痕
鉴权与协议	裸 urllib 调用（不经 SDK 掩蔽）：HTTP 200，prompt=8 / completion=1
JSON 模式	判官依赖解析成功，返回结构符合契约
Embedding 能力边界	DeepSeek 官方不提供 embedding → 按预期明确报错并**主动引导改用本地向量后端**（正确行为，非缺陷）
真实 LLM 端到端	12/12 成功，96 次调用，0 重试 / 0 截断；tokens prompt=30224 / completion=6746 / total=36970；LLM 累计 114.7s
回归测试	`test_smoke.py` 7/7（含无头 UI 全链路 AppTest，12/12）；工作台测试 11/11（含「标注与语料不匹配」的精确诊断）
降级路径	移除 Key 后重跑 → **显式降级并写入 warnings，不静默**
缓存一致性	同参数命中缓存；改 chunk_size 后自动失效重建

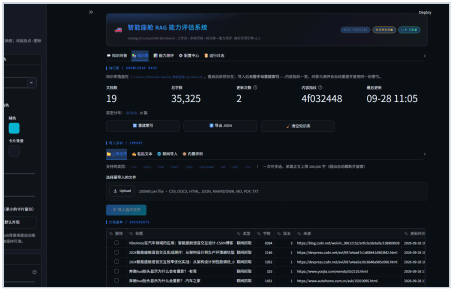
让「不确定」在界面上可见

我给 Web 界面新增了可展开面板「本次运行的真实 LLM 用量与调用健康度」，展示调用/重试/截断次数、token 拆分、累计耗时、token 上限抬升次数、判官口径；**未调用 LLM 时明确提示「指标为词法代理」**。

这个面板的作用不是好看，而是回答一个产品问题：**「这一轮到底花了多少钱、调用是否健康、我该不该信这个分数？」**——评估工具如果不暴露自身的不确定性，那它和被评估的对象一样不可信。



工作台首屏



知识库管理 · 来源与可见性



问答链路 · 回答 / 引用 / 路由判定

看界面截图时请先确认两件事：① 上面三张界面图来自 enterprise-rag-workbench 的**离线 mock 演示模式**（界面右上角有 local · mock 标识），用于展示交互与决策链路，**不代表真实模型的回答质量**；② 界面右侧出现的百分比（如引用相关度、路由判定）是**检索相关度与规则门控的信号，未经概率校准，不是「答案正确概率」**——把分数当概率正是我在座舱实验里修掉的度量失真问题，同一个错误不会在两个项目里各犯一次。

诚实边界（请一并阅读）：

- 评测集为内置样例（12 题 / 11 篇语料），**适合验证链路是否可信，不适合作为产品效果结论**。正式评估需替换为真实座舱手册与真实问题集。
- 本地向量后端为 TF-IDF+SVD（256 维），因DeepSeek 不提供 embedding，**语义召回精度受限**；如需提升需单独配置支持 embedding 的服务。
- 上下文压缩按分块逐次调用 LLM（12 题 72 次调用、4598ms/题），是耗时与 token 的主要来源。
- **无真实用户反馈、无线上灰度、无业务指标**。本项目证明的是「评估方法可运行、口径可复核」，不是「问答效果已被业务验证」。

沉淀下来的可复用方法

这个项目最大的产出不是分数，而是三条我此后每个项目都在用的规矩：

规矩 01	能一句话说清规则的模块，不要交给模型——越可复现的地方越该用代码
规矩 02	先质疑度量，再谈结论——否则只是在优化一个错的数
规矩 03	把「不做清单」写进需求文档——不写就等于默认都要做

> **如果面试时被问「这套评估体系能复用到别的场景吗」：** 可以。剥离掉座舱语料后，剩下的三层是通用的——① 指标必须声明口径（确定性 vs 模型判定）；② 失败必须显式失败，不允许静默降级；③ 文档数字必须由脚本从报告生成。这三条与行业无关，迁移到任何「AI 效果需要被证明」的场景都成立。真正不可迁移的是本项目的具体阈值（如 faithfulness 0.97），那是 12 题评测集上的结果，换语料必须重测。

下一步我会做什么

- 用真实业务文档与真实问题集替换内置样例评测集，让指标具备对外汇报资格。
- 单独配置支持 embedding 的服务，把语义召回从 TF-IDF+SVD 的 256 维升级掉。
- 做严谨的单变量消融：固定其余配置，逐项扫描 rerank / compress / chunk_size，把「端到端差异」拆成可归因的增益。

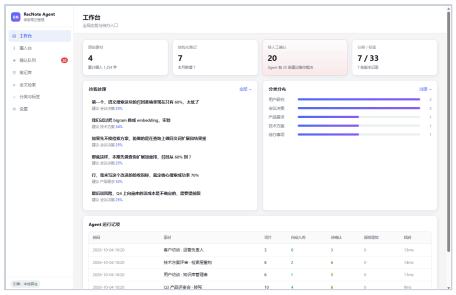
项目 02 • Agent 工作流

RecNote Agent • 把 Agent 的不确定性写成产品规则

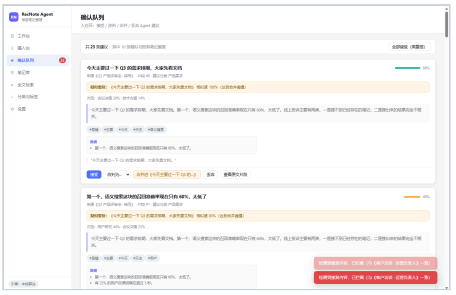
我的角色：独立项目 · 产品流程设计与原型验证（AI 辅助实现） | **规模：**7 个页面 · 5 步管线 · 17 个代码文件 | **验证：**无头 Chrome 真实点击，60+ 项断言全部通过 | **附：**56 秒产品演示录屏（另附文件）

> **对应 JD：**职责 2「将大模型、Agent、工作流等 AI 能力转化为可落地的产品解决方案」+ 职责 3「原型验证与迭代优化」。

> **配套演示录屏（另附文件，56 秒）：**按「工作台 → 摄入台 → 确认队列 → 接受一条建议 → 笔记溯源 → 版本时间线 → 全文检索 → 分类体系」的顺序走一遍完整闭环，无旁白无剪辑，1440×900 / H.264。视频里能看到 68 条待裁决建议、置信度百分比、来源片段字符位、BM25 打分与耗时，以及确认队列从 17 降到 16——**这些数字是录制时真实产生的，不是摆拍。**



工作台 · 全局态势与待办入口



★ 确认队列 · 人在环裁决



笔记详情 · 溯源与版本时间线

问题：Agent 不是「不用管就会自己变聪明」，而是「需要被约束才能用」

场景是零散、异构、无结构的录音转写文字与资料。这类任务看起来非常适合 Agent 自动整理，但我一开始就认定了一个风险：**如果 Agent 可以直接写入正式笔记，错了就没有边界。**所以这个项目的设计目标不是「自动化率最大化」，而是**「让自动化可以安全地跑」。**

产品方案：五步管线 + 人在环裁决

① 语义切片	规则边界（说话人 / 时间戳 / 标点 / 空行）+ 语义连贯性双通道，保留 charStart/charEnd 溯源锚点
② 多信号归类	关键词命中 0.4 + 分类质心余弦 0.35 + 来源先验 0.25，输出 Top-3 候选与置信度
③ 结构化提炼	TextRank 变体抽取式摘要 + 结论/待办/风险/数据四类槽位 + 领域词典优先的标签抽取
④ 相似度查重	余弦与 Jaccard 取大者，达阈值生成**合并建议**而非重复建笔记
⑤ 写入笔记库	高置信直接建笔记，**低置信降级为待确认建议**

三条核心产品规则

规则	具体设计	解决的问题
双向溯源	每条笔记能回溯到原始录音的哪一段；每个素材能列出产出了哪些笔记	AI 整理的内容无法被核对 → 等于没有信任基础
人在环	Agent 置信度不足时**不给答案**，而是把建议连同理由推给人工裁决；确认队列支持接受 / 改判 / 合并 / 丢弃	模型的不确定性被静默转成看似确定的错误结论
版本不可销毁	每次编辑 / 改判 / 合并 / 回滚都留快照，**回滚本身也记一版**	人工修正的成果被后续操作覆盖，无法追溯谁改了什么

一个我认为最有价值的设计：受限 Agent 工具集

`Agent.toolDefinitions()` 只暴露分类、草稿、记忆检索三项**受限工具——模型不能直接写正式笔记。**所有正式写入必须经确认队列。

这不是「不信任模型」的技术洁癖，而是一个产品边界判断：**把模型放在「提建议」的位置，把「落库」的决定权留在人手里**，这样即使模型表现不佳，系统也不会损坏用户数据。反过来，如果让模型直接写入，那么每一次模型失误都是不可逆的数据损失。

验证：60+ 项断言跑通，过程中发现并修复 6 个缺陷

验证方式是**无头 Chrome 真实点击**，不是接口mock。覆盖主链路 60+ 项断言与去重合并专项，包含合并、回滚、软删恢复、导入导出往返、刷新持久化等路径，全部通过。

缺陷	影响
页面事件绑定失效	核心交互不可用——自动化测试的价值就在这里
归类置信度恒低	所有笔记都进人工队列，自动化率归零，流程形同虚设
裁决非幂等	重复点击产生重复写入，是数据一致性的典型雷区
标签抽出整句	标签失去检索价值
跨词边界噪声	抽出「是搜」这类无意义标签（来自「于是搜不到」），暴露了中文分词的固有难点
去重从未触发	看似有查重能力，实际从不生效

> **诚实边界：**本项目**不宣称**完成 ASR（录音转写不在范围内），也**不宣称**开放式自主 Agent。纯前端零依赖实现，localStorage 持久化，数据自持可导出。中文检索用 bigram 建索引 + BM25 变体，不依赖分词器。LLM 增强失败时保留本地算法结果，不会阻断摄入或绕过确认队列；API Key 仅保存于浏览器会话，会话结束后不保留。

技术取舍：几个看起来是实现细节、实际是产品决策的地方

标签抽取策略	用**领域词典优先 + 虚字组合过滤**，bigram 仅作兜底——避免抽出「是搜」（来自「于是搜不到」）这类跨词边界噪声。分词方案选错了，标签就失去检索价值，这是链路会静默失效的地方
检索权重	标题权重 ×3、标签 ×2。中文检索不依赖分词器，靠 bigram 建索引 + BM25 变体实现
写操作入口	所有写操作强制经 Store 静态方法，版本记录完整率 100%——把「版本不可销毁」从设计原则变成架构约束，而不是靠人记得
三层记忆	支持会话、任务和长期偏好三层，具备来源、TTL 与删除能力；模型侧的受限工具集只暴露分类、草稿、记忆检索三项
LLM 接入降级	请求具备超时、JSON 结构与分类白名单校验，失败时保留本地算法结果，不阻断摄入也不绕过确认队列

> **这个项目教会我最重要的一件事：**「人在环」不是一个 UI 组件，而是一整套数据约束。要让人工确认真正有意义，被确认的对象必须**自带理由、置信度、来源和可回滚记录**——否则人只是在盲审，效率低且同样不可信。所以我在设计时把这四项写进了数据结构，而不是等做界面时再补。

项目 03 • 0→1 产品

暖心护 • 家庭协作小程序

我的角色：独立项目 · 需求设计、原型与版本验收（AI 辅助实现） | **技术栈：**微信原生小程序（零依赖）+ 云函数 | **验证：50/50 自动化测试通过**（v4.2.0 的 33 项 + 4.2.1 新增 17 项）· 6 视口渲染验收

> **对应 JD：**职责 2「产品从 0 到 1 的构建与上线流程」+ 职责 3「原型验证与迭代优化」。这一篇展示的是收敛能力和验收纪律。



家人端首页 · 390×1040



健康页 · 数值单位与趋势



服务页 · 卡片、筛选与价格



大字模式 · 320×1100

问题：养老场景的 AI 化不能一步到位

早期是一个泛化的养老原型。我做的第一个产品决策不是加功能，而是**收敛**：用 7 个场景、15 页原型和 4 项范围排除，先把边界确定下来，再收敛到四个模块——**家庭资料、待办、私信、SOS**。不同版本之间不重复计数。

产品方案：把「不做」写进需求

收敛后的四模块	家庭资料（成员档案与可见性）、待办（服务预约与状态流转）、私信（家庭内沟通）、SOS（求助演练）
明确的排除项	健康数据的医疗级解读、支付与商业闭环、社区内容分发——早期原型均已排除并写明理由
权限设计	家属视角隐藏添加入口并禁用用药确认；视角与大字设置跨页面保留；个人空间数据隔离
AI 助手	能读取当前心率记录并**标注来源**；对停药等超出能力边界的问题**主动返回能力边界**而非编造

「助手对停药问题返回能力边界」这一条是我特意设计的：**在养老这类高风险场景里，一个知道自己不知道什么的 AI，比一个什么都敢答的 AI 有用得多。**同样的原则也出现在 RAG 工作台的低置信拒答里。

验收：50 项自动化测试 + 6 视口渲染 + 浏览器真实流程

验证方式	结果
`npm run verify`	**50/50 通过**（v4.2.0 的 33 项 + 4.2.1 新增 17 项，**不是全页面或全仓覆盖率声明**）· 覆盖邀请码一次性与 24h过期、SOS 多方分别确认、幂等去重、断网重试、成员撤销后旧回执不重放、语音访问与删除、容量边界（普通写850KB / SOS 预留 1MB）、待删除文件不可发送或播放
`npm run check`	通过 JS 语法、JSON、12 个页面文件完整性、事件处理函数、图片引用、云函数共享代码一致性检查 · 源码及素材约 331.4 KiB
页面级回归	预约日期不能被首页日期文案覆盖；加载共享数据后仍能提交成功
6 视口渲染	1440×1200 桌面 / 390×1040 移动端 / 320×980 窄屏无横向溢出 / 健康页 / 服务页 / 320×1100 大字模式
浏览器真实流程	录入心率 76 → 健康页与首页同步显示新读数且刷新后保留；搜索无结果与保洁筛选 → 预约 → 成功页与订单页回看 → 取消；活动报名与满额禁用；保存含 HTML 字符的动态按文本展示；求助演练取消与离开页面取消

诚实边界（这一节我写得比较细）：

- **未部署**：没有真实 AppID，未在微信开发者工具完成编译、真机预览或上传审核；未部署两个云函数、未配置数据库安全规则、未做真实双设备冲突测试。云函数使用 SDK 模拟对象做契约测试，**模拟事务测试不能替代真实并发与安全规则验证。**
- **未测量**：真实设备启动时间、内存、线上可用性、真实用户成功率。
- **不代表已上线**：本项目证明的是「需求收敛与本地验收是可验证的」，不应写成已上线、已服务真实用户或已取得业务指标。
- **适老化未完成**：键盘可 Tab 到达按钮并有可见焦点，但未做完整键盘遍历、真实读屏测试、200% 缩放与真机适老验证，**不能宣称全面通过 WCAG。**求助演练路径不拨号、不获取定位、不真实通知。

0→1 过程中修复的适配器问题

浏览器适配器的全局变量回退、textarea 自闭合解析、倒计时重建节点问题、预约页面日期字段冲突。**浏览器适配器仅用于开发演示，不是微信兼容性认证**——这类「工具的适用范围」我习惯在文档里写死，避免后续被当成已验证结论。

版本演进：从泛原型到 4.2.1 开发修复包

阶段	我做的决策	验收方式
泛养老原型	场景过宽、目标不清，养老 / 社区 / 护理内容混杂，无法定义成功标准	判断：不可验收，需求返工
7 场景 → 15 页原型	把候选场景铺开逐一评估，用原型把「看起来合理」变成「可讨论的具体界面」	逐场景评审，记录排除理由
收敛到 4 模块	家庭资料 / 待办 / 私信 / SOS；明确排除健康医疗级解读、支付闭环、社区内容分发	4 项范围排除写进需求文档
家庭协作版	补齐邀请、撤销、可见性及异常处理；引入家属视角只读与大字模式	33 项自动化测试 + 6 视口渲染
4.2.1 开发修复包	按交付前审查报告修复 F01–F06（2 阻塞 / 3 高 / 1 中），逐条补回归；新增 17 项测试至 50 项；产出带每文件 SHA-256 清单的可部署源码包	**尚未部署**（线上仍 4.2.0）；三账号双平台真机、云权限拒绝测试、备份恢复演练待验收

版本之间不重复计数——这是我做迭代记录时给自己定的规则：同一个功能在 v1 有原型、v2 有测试，不该被当成两份能力。**把迭代写成「决策 + 验收方式」而不是「功能列表」，才能看出收敛能力。**

这个项目让我确认的产品判断

- **适老化的第一优先级不是功能，是「不误触」和「可读」。**大字模式不是把字号放大就行，我在 320px 视口下重排为双列→单列；家属视角直接隐藏添加入口并禁用用药确认，而不是弹个提示再让人自己小心。
- **「不做什么」比「做什么」更能体现产品能力。**7 个场景收敛到 4 个模块，靠的是明确写出排除项和排除理由，而不是克制。
- **演示环境的能力边界必须自己先说清楚。**求助演练路径不拨号、不获取定位、不真实通知——这些我在文档里写明，避免他人误以为已具备真实求助能力。

APPENDIX

其他作品索引与工程化规范

未展开详述的项目，用可验证产物与链接代替篇幅。

本次提交索引（简历项目 ↔ 仓库 ↔ 入口 ↔ 验证范围）

简历中的项目	对应仓库	入口	本次可验证范围
企业级 RAG 能力评估工作台	`intelligent-cockpit-rag`	本册第一篇；站点 rag-workbench 案例页	11 篇语料 / 12 题；真实 LLM 全链路 12/12；指标口径与复现命令
（同上：界面与知识治理）	`enterprise-rag-workbench`	站点 rag-workbench 案例页	默认 mock 演示；知识版本、FAQ 与图谱操作。**不作为真实模型能力证据**
RecNote Agent	`recnote-agent`	本册第二篇 + 56 秒实跑录屏	60+ 项无头 Chrome 断言；五步管线与受限工具集
企业 AI 案例图谱 FDE Atlas（本版新增为旗舰项）	`fde-atlas`	GitHub Pages 在线可访问；`styleguide.html` 可视设计系统	**61 项单元测试 + 59 项浏览器集成测试全绿**；127 条案例 / 中英双语；自检拦下过 6 个「单测全绿但功能是坏的」缺陷。**独立核验为 0**，已标明
暖心护家庭协作小程序	`warmcare-family-care`	本册第三篇 + 7 份验收文档	**50/50 自动化测试通过**（v4.2.0 的 33 项 + 新增 17 项）；F01–F06 六个阻塞/高/中缺陷已修复并有回归覆盖；源码含每文件 SHA-256 清单。**4.2.1 未部署**，线上仍 4.2.0
Z1Space（团队项目）	`zhihu-hackathon`	附录 + 站点 z1space 案例页	我负责产品机制与接口契约（PRD 78KB / 5 类 API / 22 项源码级评审），实现由队友承接

上表两处容易误读：① 暖心护的 50 项是**本地内存级测试 + 模拟器核对**，能证明权限与状态机的边界处理正确，**不能替代三账号双平台真机验收**——仓库主动列出 6 个当时未修复的缺陷（含 2 个阻塞级）并给出复现步骤，这是刻意的：在安全攸关场景里「已验证的边界」比「零缺陷」更可信。② FDE Atlas 的 127 条案例**独立核验数为 0**，README 与本文都列为待改进项；它证明的是信息整理与可核对性工具的设计能力，不表示作者完成过这些企业交付。

为什么把 FDE Atlas 提为旗舰项——它拦下过「单测全绿但功能是坏的」：它的价值不在 127 条案例，而在自检机制发现过一批**单测查不到**的缺陷，而当时单测全部通过：

· `search.js` 读 `detail.detail_score` 而 `buildIndex` 输出 `detail.score` → **分数差恒为 NaN，排序静默退化成只按 id 排**（单测比的是字符串字段，差为 0 时同样落到 id 分支，看起来正常）

· 三个 CSS 文件以 `/**` 开头 → 浏览器**整份丢弃样式表**（没有任何 CSS 语法检查能发现）

· 顶层 `await` 写在普通函数里 → SyntaxError，**整个文件静默不执行**（该文件不被任何单测加载）

· 事件绑定到不存在的元素、类名用了没定义、测试页路径写错 → **点击没反应 / 页面没样式 / 测试页打不开**，均不报错

修完后每一条都补了回归断言。**单测全绿不等于功能可用，得有一层专门找「安静地坏掉」的检查——这是我认为最能代表产品经理能力的一段。**

其他支撑作品（可按需追问）

作品	可验证产物	诚实边界
Agent Blackboard 多智能体协作系统	黑板架构 + 乐观锁冲突治理 + 能力路由 + 依赖拓扑排序 + 实时事件流；`github.com/Ricardo-ye/agent-blackboard`	性能数据来自 2026-09-06 历史报告；新增实现与发布设计不等于新版性能或生产容量
Enterprise RAG Workbench 工程化参考服务	生产导向的 FastAPI 参考实现：分层路由、并行检索、质量闸门、Prometheus 指标、SLO；含单元/集成/安全/UI/E2E 验收与泛化数据集；`github.com/Ricardo-ye/enterprise-rag-workbench`	默认 mock 模型与 hashing-v1 embedder 为确定性离线实现，不等于真实模型效果
bili-note 视频与图文笔记技能	SKILL.md 22,865 B · scripts/ 10 个脚本 · 17 个代码文件 / 28 个测试用例 · 覆盖 7 个环节；已作为可复用技能发布	依赖 B 站页面结构，页面改版会导致解析失败
技能描述优化器 单文件 Web 应用	单文件 31,574 B / 内联逻辑 14,478 字符 / 26 个交互控件 / 7 个功能点；原始描述 → 三档优化 → 评分诊断 → A-B 对比 + 触发命中测试；双引擎可选	无测试用例（「中」强度）；离线引擎输出偏模式化
Z1Space 双人协作（团队项目）	我在 Z1Space 负责产品机制与技术方案：Agent 提供候选与理由、关系由真人确认；将 pending 状态、证据约束与失败回退写成可检查规则，围绕 5 类 API 明确字段、状态与验收条件	队友负责实现；无真实用户反馈、推荐效果或线上灰度结论

工程化规范（三个项目共用）

单一事实来源	data/reports/*.json 为报告唯一来源，文档只做转述并附复现命令——文档数字若非脚本生成，一定会腐化
不静默降级	任何能力缺失都显式写入 warnings 并在界面提示，不让「看起来成功」掩盖真实失败
口径分离	确定性指标与 LLM 判官指标分口径呈现；对比时明确说明是否可逐行对比
不写清单	每个需求文档必带「不做清单」，不写就等于默认都要做
范围隔离	浏览器适配器仅用于开发演示，非微信兼容性认证；模拟事务测试不替代真实并发验证

为什么我的材料里「诚实边界」写得这么多

把话说明白：我认为这册最值得看的地方，不是三个项目的功能列表，而是三处「主动暴露问题」。① 第一篇把「界面百分比不是答案正确概率」写在截图旁边而不是藏进脚注——把检索得分当概率正是我在评测里修掉的失真问题，同一个错不该在两个项目里各犯一次。② 第二篇公开了 demo 截图中那张答非所问的样例（问「数据如何隔离」答「系统架构分层」），并写明它**不是**能力证据。③ 附录里两个旗舰项目的未验证项与已完成项并列在同一张表右侧——包括暖心护「4.2.1 未部署」与 FDE Atlas「127 条案例独立核验为 0」。

如果这份材料里所有数字都好看，评审反而无从判断我是否真的做过。**留出可查的边界，别人才知道哪些结论可以引用。**

因为 AI 产品最容易交付、也最容易被复用的一类东西，就是**过度承诺。**一个把mock 结果说成真实效果、把模拟测试说成并发验证的作品集，在技术面试里一问就穿帮。所以我给自己定了规矩：**每个结论都要能回答「这个数字怎么复现、适用范围是什么、什么情况下它不成立」。**这份作品集里所有的限制声明都不是谦虚，是可验收性的一部分。

提交前自检清单（如实标注，不隐藏）

项目	已完成并可当场复核	尚未验证 / 不成立
RAG 评估工作台	指标口径与复现命令、真实 LLM 调用记录、9 个问题的修复与回归、耗时口径拆分	真实用户效果；12 题之外的大样本泛化；单变量消融尚未做
RecNote Agent	60+ 项无头断言、6 个缺陷的复现与修复、56 秒实跑录屏	ASR 转写、开放式自主 Agent、多用户与真实数据规模
FDE Atlas	61 项单元测试 + 59 项浏览器集成测试；127 条案例经 schema 三层校验；`styleguide.html` 设计系统；GitHub Pages 在线可访问	127 条案例**独立核验为 0**；14 组架构模板化；11 条来源链接复用
暖心护	需求收敛与排除理由、**50/50 自动化测试**、F01–F06 六缺陷修复与回归、7 份验收文档、源码包 SHA-256 清单	**4.2.1 未部署**（线上 4.2.0）；多账号真机、云权限拒绝、备份恢复演练均未做；无真实定位与通知
本次提交材料	本册全部数字可回溯到项目内报告文件；站点与仓库链接均可访问	`enterprise-rag-workbench` 的 mock 截图中有一张答非所问样例（问「数据如何隔离」答「系统架构分层」），本册不把它当作能力证据

联系方式13516594489 · 2951026785@qq.com ｜ **个人网站** myricardo.online（完整案例拆解与界面设计稿） ｜ **GitHub** github.com/Ricardo-ye

文档版本 2026-10-22 · 第 2 版 · 全部数字来自项目内真实运行报告，无模拟或占位值 · 已按公开仓库逐项核对