Complywerk
欧盟 EPR / GPSR 合规运营平台

面向机构负责人与合规主管的技术与业务深度说明

Complywerk 深度手册

每一条能力都标注代码出处 · 未实现的写在 Roadmap,不混进能力页

这本手册回答四个专业买家真正关心的问题:
会不会算错?出了事能不能自证?数据安不安全?我要走的时候怎么办?

读者	性质	版本	配套
机构老板 / 合规负责人	能力均可溯源验证	2026 · 深度版	另有《上手教程》入门版

速览 一页看完全部深度

深度速览:8 个「别家不做、我们做了」

每一条在后文都有专页展开,并标注代码文件出处——你可以在演示或私有化部署里逐条验证。

① 按产品逐个求值,不按客户一刀切

「产品A有包装 + 产品B卖德国」**不会**拼出德国包装义务。条件树对每个产品独立求值。

→ 第 3 页

② 判不了的显式说,绝不静默判「无义务」

未被规则覆盖的 国家×合规流 单元格显式产出**警告**;连「引擎表达不了的法条」都留档原因(如德国 VE 投放量门槛)。

→ 第 3/11 页

③ 每个数字带来源标签,审计场景可自证

申报数值三档来源(实测/规格书/估算),估算值在导出文件头**显式披露**——德国 Abmahnung 场景数据可自证。

→ 第 5 页

④ 审计日志连数据库管理员都改不了

防篡改在 **DB 触发器层**(UPDATE/DELETE 一律异常),不是应用层约定。

→ 第 5 页

⑤ 租户隔离是数据库策略,不只是代码习惯

PostgreSQL RLS + 受限角色 + 每请求 GUC:代码层哪怕写漏 where,DB 层默认**拒绝返回 0 行**。

→ 第 6 页

⑥ 授权书与客户退场是状态机,不是备注字段

mandate 四态(到期自动预警/形式要件硬闸/终止派生任务);退场三态含 **nil-return 尾巴期**——注销拖尾期继续报零,这是机构最常漏的责任。

→ 第 9/10 页

⑦ 法律细节编进数据模型

LUCID 注册标记「客户本人必办」(VerpackG §9 不可代办);GB/NI 温莎框架双轨;英国半年报频率引擎级支持。

→ 第 11 页

⑧ 费率预估:有官方数字用官方,没有就明说没有

UK 官方费率(OGL 授权)+ RAM 红黄绿 what-if;德国市场参考带;**法国不显示数字——因为不存在可靠的公开数字**。诚实本身是能力。

→ 第 12 页

验证方式 —— 本手册所有能力页均附「代码出处」脚注(灰色等宽字块)。评估时可要求演示对应行为,或在私有化部署中直接查验源码与测试(全仓 1459 项自动化测试,见第 13 页)。

1 会不会算错 / 漏报 · 上

求值语义:宁可显式报「不知道」,绝不静默判「没义务」

合规系统最危险的错误不是算错,是**漏**——静默地告诉你「这个客户没事」。Complywerk 的求值引擎在三个层面堵死假阴性。

按产品逐个求值

规则条件树对**每个产品独立求值**:必须「单个产品整组满足」才触发义务。客户有产品A(带包装、只卖法国)和产品B(无包装、卖德国)时,「有包装+卖德国」**不会**被跨产品拼出来误报德国包装义务。按产品粒度的义务(如注册号挂靠)对每个命中产品各发射一条并携带 product_id。

> 代码出处
> packages/rules/src/evaluate.ts — evaluateRule 先 products.filter(p => evaluateCondition(conditions, client, p.facts)),per:'product' 逐产品发射

覆盖缺口显式兜底

系统按产品属性推导「应该适用哪些合规流」(有包装→packaging、电子→weee、含电池→battery、GPSR 恒适用),再对照已发布规则——**任何未被覆盖的 国家×流 单元格都显式产出警告卡**,出现在工作台「规则未覆盖客户」与客户 360。你看到的不是「没义务」,而是「此处无规则,请补规则或人工核实」。EU 域规则按 EU27+NI 展开逐国对账。

> 代码出处
> packages/rules/src/coverage.ts — computeCoverageGaps 纯函数,reason:'no_rule' 单元格;EU27_MEMBERS 含 NI

求值结果走 diff 确认流,不静默改库

规则重算永远输出「拟议变更」四桶:**新增 / 移除 / 变更 / 不变**。变更桶对规则版本、所需文档、频率、截止规则、提前天数五个字段逐项留痕(from→to)。专员确认后才落库生效——系统绝不因为规则升版就悄悄改动已确认的义务。

> 代码出处
> packages/rules/src/evaluate.ts — diffObligations:added/removed/changed/unchanged;ObligationFieldChange {field, from, to} 五字段留痕

> **规则库规模与权责边界** —— 平台规则 33 个 key、库内 67 行多版本共存,覆盖 10 国 EPR(DE/FR/ES/IT/UK/PL/NL/AT/BE/SE × 包装/WEEE/电池)+ EU 级 GPSR、CE-DoC、化妆品 RP。每条规则带官方来源引用与 `review_status` 权责标记: `ai_researched` (AI 依官方来源研究)→ `expert_verified` (持牌人员核对)——对真实客户出法律结论前的核对责任**显式建模**,不假装 AI 研究等于法律意见。

> 代码出处
> packages/rules/src/rules/index.ts — count(distinct key)=33 / count(*)=67 与 README 对账;types.ts review_status

1 会不会算错 / 漏报 · 下

申报数据的七道合理性校验

数据在**导出成官方申报文件之前**先过校验引擎:3 个 error 直接拦下文件,4 个 warning 提示复核——「导出一份错误的官方文件」比「不导出」伤害更大。

校验码	级别	语义
negative_grams	● error	负克重——直接拦截导出
empty_total	● error	总量为零——拦截导出
missing_required_material	● error	缺必报材料类别——拦截导出
abnormal_large_value	● warning	单材料超 100,000 kg 阈值——提示人工复核
period_deviation	● warning	同比偏差超 50%——提示解释波动
period_deviation_estimated	● warning	偏差且数值为估算来源——双重提示
nil_after_activity	● warning	此前有量、本期报零——防误报零

代码出处
`apps/app/src/server/submission-export/plausibility.ts — 3 error + 4 warning 码;errors 阻断导出路径`

「引擎表达不了的,如实留档」

有些法条依赖系统没有的事实——例如德国 VE 完整性声明只适用「上年投放量超门槛」(轻包装 30t / 纸 50t / 玻璃 80t)的生产者。引擎没有投放量门槛事实,**就不发射这条义务,并把原因写进规则头注释与 changelog**,由人工判断。同类:ElektroG 纯 B2B 生产者的申报分叉。

行业常见做法

- ✕ 门槛判不了 → 全量发射,让用户自己删(误报噪音淹没真义务)
- ✕ 或干脆不建模,静默没有这条(假阴性)

Complywerk 的做法

- ✓ 不发射 + 原因留档在规则注释与版本 changelog
- ✓ 「系统能判什么、不能判什么」对用户透明——这正是专业机构需要的责任边界

代码出处
`packages/rules/src/rules/de.ts — 头注释与 changelog 明写 VE 门槛不可表达故不发射;B2C/B2B 分叉同`

2 出了事能不能自证

数据可信链:从录入到导出,每一步可回溯

审计、平台问询、德国 Abmahnung(竞对警告函)场景下,机构最需要的不是「数据在系统里」,而是**「数据怎么来的、动过没有」说得清**。

1. **三档数值来源标签。**每个申报数值标 `measured`(实测)/ `spec_sheet`(规格书)/ `estimated`(估算);未标注的**缺省按 estimated**——宁可标低可信度,不留空白。含估算值的英国 RPD 导出文件头部自动追加披露注释(「Value sources: N of M estimated」),与实际行数严格一致。

2. **审计日志 DB 触发器防篡改。**`audit_events` 表上有数据库级触发器:任何 UPDATE / DELETE 一律抛异常——这不是应用层约定,是数据库物理层强制,**本地超级用户也改不了历史**。谁在什么时候确认了哪条义务、终止了哪份授权书,写入即定格。

3. **官方规范级导出。**英国 RPD 导出按 gov.uk 2025 file specification 的 15 列格式与受控枚举(organisation_size S|L、packaging_activity SO|PF|IM|SE|HL|OM、packaging_type HH|NH|CW|OW|PB|RU|HDC|NDC|SP),CRLF 行尾;合理性 errors 直接拦下文件;导出动作本身入审计。

4. **证据包的诚实机制。**Amazon 证据包(DE/FR)导出 ZIP:注册号清单 + 证书文件 + **GAPS.txt 缺口清单**——过期文件标 `!! EXPIRED`、缺失标 `!! MISSING`,绝不静默省略;无缺口时也写「No gaps」自证完整。文件字节经 sha256 与金库版本三方一致校验。

5. **交付物同源。**客户月报的「缺材料清单」与健康度引擎**完全同一口径同一函数**——不会出现「报告说齐了、系统说缺」的两张嘴。

> 代码出处
> `packages/db/src/schema.ts FILING_VALUE_SOURCES · packages/db/drizzle/0008_audit_append_only.sql(BEFORE UPDATE OR DELETE … RAISE EXCEPTION)· apps/app/src/server/submission-export/uk-packaging.ts · evidence-pack(GAPS.txt/sha256)· routers/reports.ts(computeMissingMaterials 同源)`

3 我的数据安不安全 · 上

租户隔离:数据库层策略,不是代码层习惯

多租户 SaaS 的隔离通常靠「每条查询都记得加 where tenant_id」。Complywerk 在这之外加了一道**数据库强制层**:代码写漏了,DB 也不给数据。

三层隔离:应用守卫 → 请求级 GUC → 数据库 RLS 默认拒绝(图中术语保留英文,便于转发给技术评估人)

a **受限运行角色。**应用运行时使用 `complywerk_app` 角色:NOSUPERUSER、NOBYPASSRLS——即使应用被攻破,连接本身也无越权能力。迁移/运维走独立 owner 连接双轨。

b **默认拒绝。**22 张业务表全部启用 RLS;策略要求 `tenant_id = current_setting('app.current_tenant')`——GUC 未设置时**返回 0 行**,而不是全部行。忘了设租户 = 什么都看不到,故障安全方向正确。

c **新表红灯机制。**刻意不授 DEFAULT PRIVILEGES:未来新表若漏配 RLS,自动化测试(「22 张表全启用」断言)直接红灯——隔离纪律由测试强制,不靠记性。

> 代码出处
> `packages/db/drizzle/0018_enable_rls.sql`(角色/22 表策略/默认拒绝/红灯注释)· `apps/app tRPC enforceSession→withTenant()` 事务级 `set_config`

3 我的数据安不安全 · 下

认证与权限:细到按钮,严到时间侧信道

认证栈

1. **argon2id 密码散列**(Rust 原生实现),参数显式钉住不随库默认漂移;设密码最大长度防超长输入拖垮散列器。
2. **时间侧信道防账号枚举。**登录邮箱不存在时,仍对一个固定假散列做完整 argon2 验证——「账号存在」与「密码错误」在响应时间上不可区分。
3. **TOTP 两步验证带防重放。**`totp_last_used_step` 要求严格递增——同一个 6 位码不能用两次;恢复码 argon2id 散列存储、一次性消费、明文只在生成时出现一次。
4. **锁定策略。**登录与 TOTP 均 5 次失败锁 15 分钟;邀请/重置走一次性 token 真实邮件通道并入 `email_log` 留痕。

> 代码出处
> `apps/app/src/server/auth/password.ts(argon2id/假散列)· auth/service.ts(TOTP_MAX_FAILURES=5 / LOCK 15min / totp_last_used_step 严格递增)· invite-email.ts`

权限:按钮级,拦在点击前

四角色(owner / admin / specialist / viewer)的能力差异做到**按钮级**:无权限的写动作直接灰置并带原因提示(「只读角色不可执行」),而不是「点了再报 403」。服务端同一阶梯二次校验——UI 灰置是体验,服务端拒绝是边界。破坏性动作(如 mandate 终止)独立更高门槛(admin 专属)。

> 代码出处
> `apps/app/src/lib/role-context.tsx + capabilities.ts(canWrite/writeReasonKey,20 个写动作面门控)· 服务端 roleProcedure 阶梯`

> i **合规配套** —— GDPR 信任中心(/trust)公开:子处理器清单、数据驻留、责任边界(系统能力 vs 持牌法律判断的分界)、DPA 要点。演示环境与生产环境行为一致,可实地核验。

4 你们跑路了怎么办 / 我要走怎么办

数据归属与退出路径:如实写清现状

评估 SaaS 时最少被回答、又最该被回答的问题。我们按现状如实列出——**包括还没做完的部分**。

1. **数据归属你。**合同与信任中心明确:租户数据归机构所有,我们是处理者。数据库为标准 PostgreSQL——不是私有格式,`pg_dump` 即可得到完整、可迁移的副本。

2. **现有导出面(今天就能拿走的)。**客户/产品 CSV(与导入同构,可回导)、客户月报 PDF、英国 RPD 申报 CSV、证据包 ZIP(注册号+证书原件)、文档金库原文件(上传原字节 sha256 可验)。

3. **私有化部署是一等公民,不是销售话术。**Docker Compose 单机即起(app+db+daemon,镜像 ~223MB),数据从第一天就在你自己的服务器上;SaaS 与私有化同一代码线,不是阉割版。对数据主权敏感的机构可以直接从私有化开始。

4. **诚实缺口:自助式一键全量导出尚未做**(在 Roadmap)。当前全量迁出路径:私有化租户 = pg_dump 自取;SaaS 租户 = 按请求由我们出全量导出(标准格式)。不写「随时一键带走」的空话——写我们真的能兑现的。

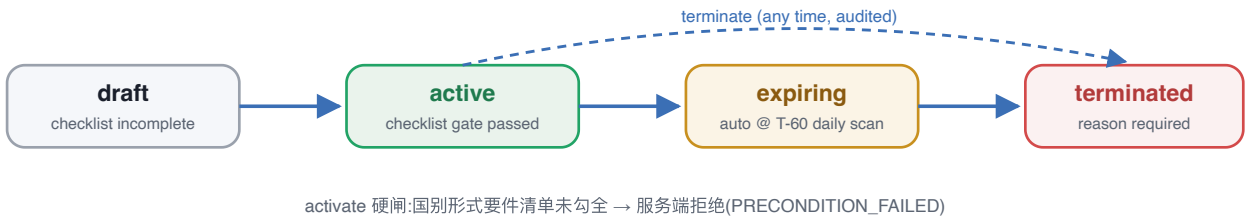
> 代码出处
> `docker-compose.yml + DEPLOY.md(私有化)· server/importer.ts(CSV 同构)· submission-export/ · evidence-pack · /trust` 信任中心

✓ **为什么敢这么写** —— 这一页的存在本身就是产品哲学:能力页只写已实现并有测试覆盖的;没做的进 Roadmap 并注明路径。你在评估其他供应商时,不妨拿同样四个问题问他们。

5 生命周期建模 · 上

授权书(Mandate):一等实体 + 状态机

授权书是机构代办业务的**法律地基**——它过期、形式要件不全、终止不通报,都是真实责任事故。Complywerk 把它建成带状态机的一等实体,不是客户档案里的一个附件。



activate 硬闸:国别形式要件清单未勾全 → 服务端拒绝(PRECONDITION_FAILED)

Mandate 四态状态机;expiring 由每日扫描按 T−60 自动流转,不靠人记

1 **形式要件是硬闸不是备忘。**德国 WEEE 授权书(ElektroG §8:书面、德语、明确设备类别范围)与包装授权书(VerpackG §35:德语书面、单一授权代表)各有配置化清单;**必选项未勾全,服务端直接拒绝激活。**勾选是留档动作,法律判断仍归持牌环节——边界写在产品里。

2 **到期不靠人记。**active 且距到期 ≤60 天自动转 expiring;提醒按租户档位([30,14,7,1] 天,可配)进 digest。工作台有全租户到期列表。

3 **终止有连锁动作。**终止必填原因、入审计,并**幂等派生**「通知注册机构撤销授权」任务(以 mandate_id 锚定,重复终止不重复派生)——把「忘了给 stiftung ear 发通报」这类事故从流程里消掉。

代码出处

`apps/app/src/server/mandates-engine.ts(MANDATE_EXPIRING_WINDOW_DAYS=60)· lib/mandate-checklists.ts(法条出处内嵌)· routers/mandates.ts(activate 闸/terminate 派生)· packages/db schema mandates/tasks.mandate_id`

5 生命周期建模 · 下

客户退场:注销的「尾巴期」才是责任高发区

客户不续约了,义务并不立即消失——德国登记处在公示离场日之前**仍要求继续申报(报零)**;LUCID 注销不等于双元系统合同解约;法国错过 10/31 挂号信窗口就多付一整年。这些「尾巴」是机构最常踩的坑,我们把它做成了状态机。

退场三态;leaving 期间申报任务照常生成并带「零申报 —」前缀,公示离场日后才停发

1 **Nil-return 尾巴期语义。**进入 leaving 后系统**不停发申报任务**,而是给任务标题加「零申报 — / Nil return — 」前缀;到 market_leaving_date 公示、状态置 closed 才停。漏报尾巴期在德国会被登记处追责——这是全流程里最反直觉、也最值钱的一条建模。

2 **国别注销清单是双轨的。**德国包装退场清单把「登记处注销」与「双元系统合同解约」列为**两条独立轨**,各含「行政确认文书已归档」核验步骤——LUCID 注销≠合同解约,漏了后者客户会继续被扣费。

3 **法国 10/31 退出窗口。**自动生成带截止日的清单任务(下一个 10/31),文案带「以合同文本为准」限定语——likely 级研究结论在产品文案里如实降级,不冒充确定。

4 **双状态防漂移 + 可撤销。**closed 自动同步客户主状态为 churned(唯一映射,不出现两个字段各说各话);撤销路径把派生任务与 nil 标记一并清理,closed 为终态不可撤。全流转入审计。

> 代码出处
> `apps/app/src/server/offboarding.ts(applyOffboardingTransition 纯函数/状态映射)· filings-engine(nil-return 前缀/停发边界)· offboarding 清单配置(DE 双轨/FR 窗口 frExitWindowDueDate 边界测试)`

6 法律细节与多语言架构

法条的「坑」编进数据模型;语言跟着交付物走

四个别家不建模的法律细节

细节	建模方式
LUCID 不可代办	VerpackG §9 要求制造商**本人**注册——该义务标 `delegable:false`,机构端与客户门户三语显示「必须由贵司本人完成」徽标。机构越权代办是法律瑕疵;白标门户不区分等于诱导越权。
GB / NI 双轨	英国规则条件 any:[GB, NI](北爱尔兰留在英国 pEPR 体系内);同时 NI 属 EU 域规则展开集(温莎框架)——**只卖北爱的客户会同时触发英国与欧盟两套义务**,系统如实双发。
半年报频率	英国大生产者半年申报:引擎级 `semiannual` 频率(H1: 1/1–6/30,H2: 7/1–12/31),期末 +93 天提交截止(对官方日历核验:H1 → 10/1)。
费用义务类型	义务类型含 `recurring_fee`(缴费型):带币种与计费方式说明、实缴金额/日期/发票文档挂接——「要交钱」在矩阵与日历是一等公民。

代码出处

`packages/rules/src/rules/de.ts(delegable:false 注释引 §9)· uk.ts(any GB/NI)· coverage.ts(EU27+NI)· types.ts Frequency semiannual · periods.ts(93 天断言在 g3-rules.test.ts)`

多语言:语言属于交付物的收件人

1. **义务名按租户语言物化。**规则内容是 {zh, en} 双语对象,在求值落库时按**租户** locale 定格成单语快照——英文机构的 282 条演示义务名零中文,不是翻译出来的,是生成时就对的。
2. **客户门户三语**(zh/en/de)零新依赖;**月报语言跟随租户**而非操作者——中文运营人员给英文客户出的报告是纯英文(这是修过真实混排 bug 后的架构决定:交付物语言归属收件人)。
3. **错误路径也是英文的。**tRPC 错误与表单校验消息走字典键、在响应边缘按请求语言翻译——英文用户在任何路径(包括报错)看不到中文。

代码出处

`packages/rules evaluate.ts(locale 物化)· apps/app/src/portal-i18n.ts(三语 PORTAL_DICT)· (print)/reports(tenants.locale)· server/app-error.ts(LocalizedTRPCError)`

7 费率预估引擎

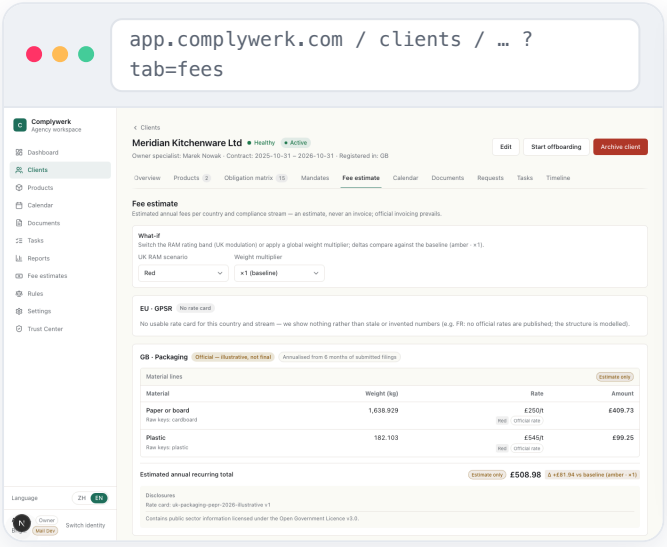
「明年要交多少钱?」——有官方数字用官方,没有就明说没有

机构最常被客户问、也最难答的问题。Complywerk 的费率引擎按**三档诚实度**回答,每个数字带来源与可信度徽标,全链路标注「预估,非账单」。

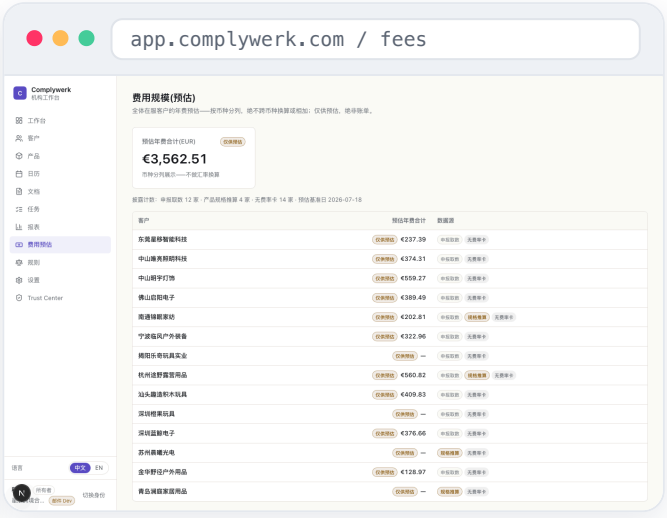
档位	内容
官方费率	英国 pEPR 处置费:8 材料 £/吨,数据来自 gov.uk(开放政府许可 OGL v3,带署名);2026–27 官方 illustrative 费率以「官方·非最终」徽标呈现,绝不冒充定案。德国 WEEE 行政官费(费用条例直载)精确到分。
市场参考	德国包装双元系统为竞争定价、无官方费率表——给出公开门户 2026 均价作为参考带,并明示「小额牌价,大宗谈判价更低,以报价为准」。
显式「无数字」	法国 Citeo 费率官方 PDF 门控、第三方转载互相矛盾 >2 倍——**我们不显示数字,并告诉你为什么**。德国纸/玻璃无可靠公开价,同样显式说明。宁可少答,不给假精确。

What–if:英国 RAM 红黄绿场景

2026–27 起英国按可回收性评级调节费率:红档 = 基线 ×1.2,2028–29 升至 ×2.0。引擎按官方绝对费率做 green/amber/red 三场景即时对比——「换可回收包装能省多少」从咨询话术变成屏幕上的 Δ 数字。机构组合视图把全部托管客户的年费规模按币种汇总(GBP/EUR 分列,不做汇率虚构)。



英国客户 · RAM red 场景:行级「Official — illustrative, not final」徽标、red 绝对费率(£250/t、£545/t)、Δ +£81.94 对比基线、OGL 官方数据署名。



机构组合视图:全部托管客户年费规模,按币种分列;每行「仅供预估」徽标 + 数据源徽标(申报取数 / 规格推算 / 无费率卡)。

> ! **边界声明(产品内同款措辞)** —— 所有金额为规划用预估:英国实际费用以 PackUK 开具的 Notice of Liability 为准;德国包装以所选双元系统报价为准。预估永远带披露徽标,系统不生成、也不冒充账单。

代码出处

`packages/rules/src/rates/`(卡数据+纯函数引擎)· `docs/rate-card-evidence.md`(每个数字→官方来源 URL→引句 1:1 对照,随代码入库)

8 工程质量基线

你看不见的部分,决定三年后还好不好用

指标	数值与含义
自动化测试	**1459 项全绿**(rules 304 / db 86 / app 1069,2026-07-18 实测,含费率引擎批次新增 189 项)。规则库「就是测试库」:每条模板规则有快照与正路径测试;规则与费率卡升版必须同步动 expected-version 棘轮测试——改数字必留痕。
类型纪律	全仓 TypeScript strict + `noUncheckedIndexedAccess`:所有索引访问强制判空——对逐材料、逐国家聚合的合规数据代码,这是把一类「漏一格」错误在编译期消灭。
数据库迁移	21 个 SQL 迁移(0000–0020)全程幂等纪律(存在守卫/先删后建),含审计触发器、RLS、双语化回填等手写迁移——可重放、可审计。
演示自检	演示租户种子末尾带**硬断言**(健康度红3/黄4/绿7、授权书数量等),任意播种日期恒成立——演示环境不会悄悄烂掉。
对抗式验收	每个功能批次经多智能体对抗评审(专门找茬:跨租户攻击/角色越权/数字保真/误导性界面),发现与修复记录公开在仓库 DECISIONS.md——包括我们自己犯过的错。

诚实工程的三个习惯

1. **偏差披露。**验收记录里未达原计划的项标 🔶 并写实际交付形态与原因——不静默降级。
2. **「够用就停」。**拒绝为炫技建模(不做汇率虚构、不做假 AI 结论);每个「不做」都有记录在案的理由。
3. **能力/结论分界。**系统输出的是「依据 X 规则 Y 版本、来源 Z」的可追溯建议;法律结论的最终责任在持牌环节——这条边界写在信任中心、写在规则元数据、也写在这本手册里。

> 代码出处
> `tsconfig.base.json · packages/db/drizzle/(21 迁移)· DECISIONS.md(全部批次验收与偏差记录)· demo seeds 硬断言`

9 ROADMAP 与边界

接下来做什么;以及我们不做什么

Roadmap(按优先序)

项	说明
法国费率数字接入	获取官方 barème PDF 后接入(结构已建模);Citeo Pro 是否入范围届时显式决策。
费率↔发票对账	「PRO 发票金额 = 申报量×费率」核对——全行业空白,费率卡是它的地基。
Eco-modulation 深化	法国 bonus/malus 项级建模;英国 RAM 评级录入直通 RPD 导出。
自助全量导出	租户自助一键全量导出(当前:私有化 pg_dump / SaaS 按请求)。
英文版深度手册	面向欧洲机构;本册架构图已用英文标注,便于先行转发。

恒久不做

汇率虚构合计 · 假装可精确复算官方开票 · 无来源的费率数字 · replace 持牌判断的「AI 法律结论」。

Complywerk

complywerk.com · hello@complywerk.com

深度不是功能数量,
是每一条能力都敢写出处。

评估建议:带着本手册预约演示,逐页要求现场验证;或申请私有化试装,直接查验源码与测试。

