GARY CHEN VIDEO NOTES

`/grill-me` 到可控 AI 开发流

从拷问需求，到 spec、ticket、TDD、review、写作

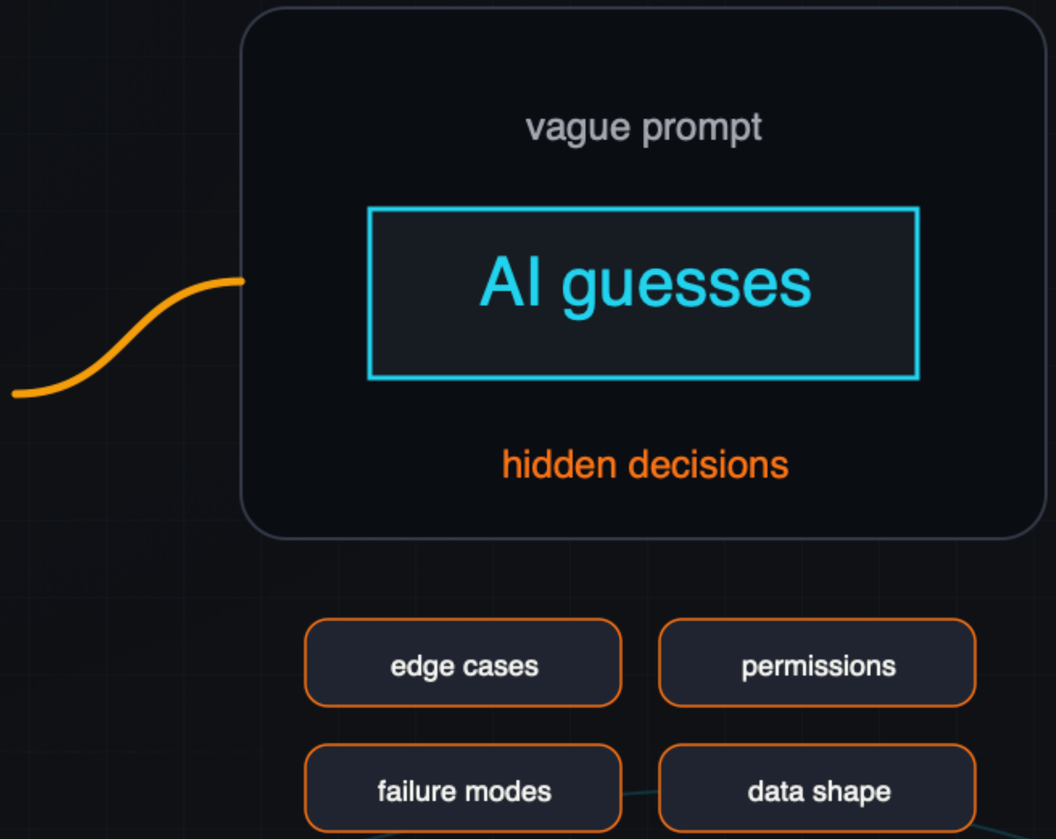
grilling - to-spec - tickets - TDD - review - writing

PROBLEM

问题不是 AI 不会写 code，而是它太会猜

模糊需求会把几百个微观决策外包给黑盒

- 防呆、断线、边界条件会变成隐形决策
- Vibe coding 的风险：AI 替你拍板，你承担维护成本
- Matt 的核心目标：把决策权抢回人类手里



TOOL SHAPE

不是接管整条流程，而是给你一盒可拼装工具

小
skill，可替换、可跳步，错误更容易被局部发现和替换
重流程

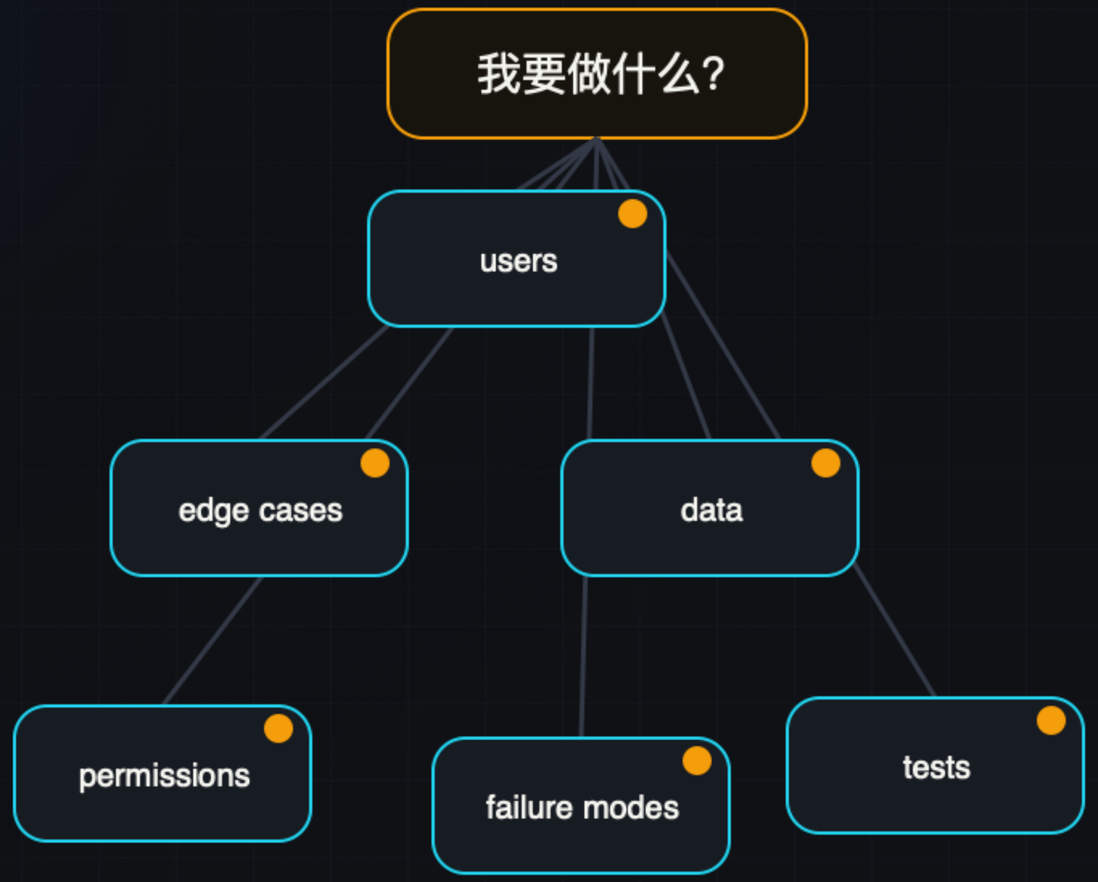
- 重流程：spec 到 implement
  全绑死
- 小工具：每个 skill
  只解决一个问题
- 关键差异：主导权留在人手里

/GRILLING

`/grill-me` 的本质：让 AI 画出你的设计树

不是让 AI 做决定，而是逼你暴露前置决策

- `/grill-me` 在当前 repo 中会运行 `/grilling`
- 每个决定会解锁下一层问题
- AI 给推荐答案；最终拍板的人还是你



TO-SPEC

`to-spec` 把共识冻结下来，但不把代码写进去

spec 记录业务决策，不缓存容易过期的实现细节

- 把当前对话和代码理解合成为 spec
- 记录 problem、solution、user stories、testing decisions
- 少放具体文件路径和代码片段，降低文档漂移

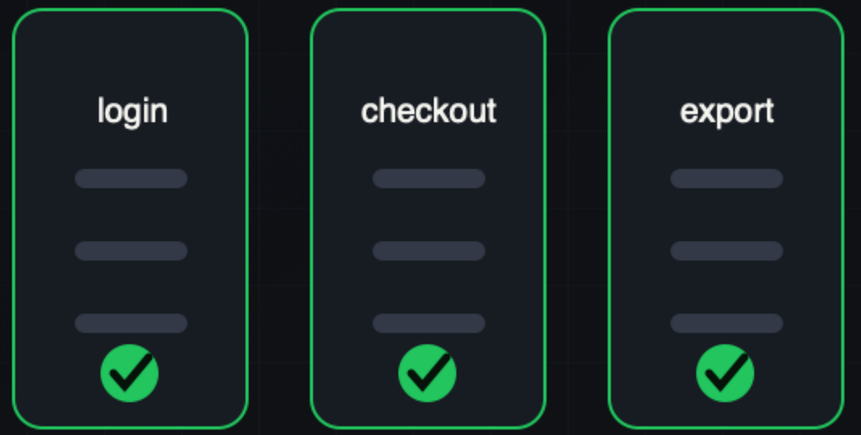
TO-TICKETS

`to-tickets` 不按技术层拆，而按可验证功能切

每张 ticket 都是一条 tracer bullet vertical slice

错误拆法

正确拆法

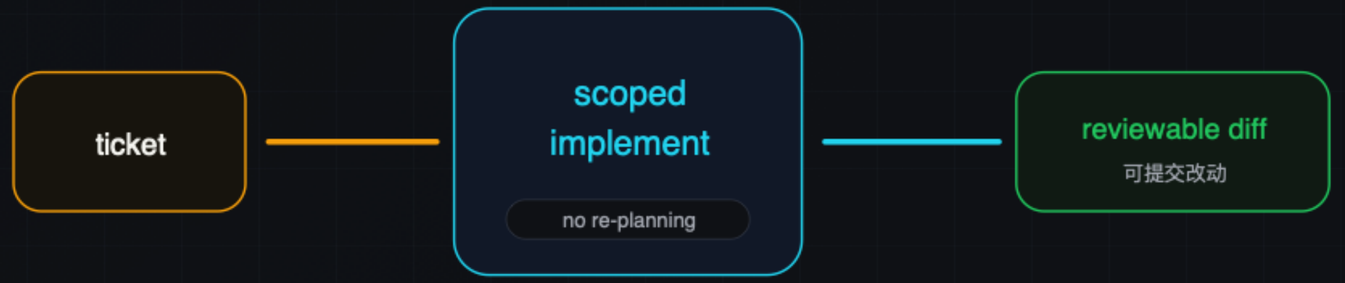


- 错误拆法：先 database，再 backend，再 frontend
- 正确拆法：一张票贯穿 schema、API、UI、tests
- 每个 ticket 标明 blocked by，形成可工作的 frontier

IMPLEMENT

`implement` 的价值在于不重新发明需求

上游决定清楚后，下游只负责把一张 ticket 推到可提交状态



- 读取 spec 或 ticket
- 在预先同意的 seams 上运行 TDD
- 频繁 typecheck 和单测，最后跑 full suite
- 完成后运行 `code-review`，再进入提交或人工确认

TDD

TDD 是降低 AI 自证正确风险的约束

先红灯，再绿灯；先定事实，再让模型写实现

- 测试必须验证 public seam，不测内部细节
- expected value 来自 spec 或 worked example
- 一次只做一个 vertical red-green slice

tautological test = fake green

CODE-REVIEW

`code-review` 分两条线审：Standards 和 Spec

一条看代码有没有坏味道，一条看它有没有做对需求

- Standards：项目规范 + Fowler smell baseline
- Spec：缺失需求、scope creep、错误实现
- 两个 review 分开跑，避免一个维度掩盖另一个维度

ARCHITECTURE

AI 最容易造出浅模块：文件很多，门却很少

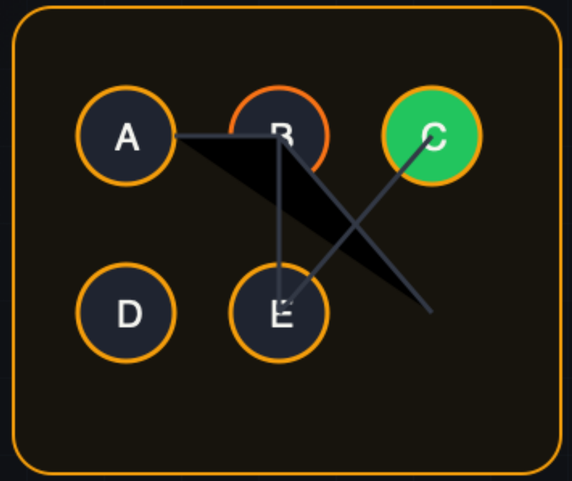
深模块把复杂性藏在小接口后面，让 AI 看得完整

- 浅模块：调用方被迫知道太多内部路线
- 深模块：一个简单 interface 包住复杂流程
- 对 AI 的意义：上下文集中，修 bug 不靠瞎猜

MAINTENANCE

架构不是一次性设计，而是每隔几天体检

`improve-codebase-architecture` 用 deletion test 识别 deepening opportunities



- 删掉模块后复杂性外溢，它可能真的隐藏了复杂性
- 删掉后更清爽，它可能只是 middle man
- 先出 HTML report + grilling，再决定是否重构

WRITING

写给 agent 的文档，默认动作应该是删

`writing-for-agents` 追求稳定过程，不是更长解释

- Context pointer：什么时候该加载这份材料
- Leading words：用高密度术语压缩行为
- Pruning：删掉 no-op、重复和分散注意力的句子

WRITING FLOW

写作也能被拆成 explore 和 exploit

先挖 fragments，再用 shape 或 beats 组织成文章

- `writing-fragments`：只收集 raw fragments
- `writing-shape`：段落级塑形，确定读者顺序
- `writing-beats`：beat-by-beat 推进，每一步先 ground 概念

TRADEOFF

Superpowers 是生产线；Matt 的 skills 是乐高积木

一个保证不漏步骤，一个让你按需要组合

- Superpowers：适合需要高约束护栏的完整流程
- Matt skills：适合模型更强、人更清楚、流程要可插拔
- 选择标准：稳定流水线，还是可组合控制面板？

OPERATING RULE

先把决定留给人，再把执行交给 AI

grilling -> to-spec -> to-tickets -> implement -> code-review

writing-for-agents as reusable discipline

- `grilling` 暴露决策树
- `to-spec` 固化共识
- `to-tickets` 切成可验证 vertical slices
- `tdd + code-review` 让反馈闭环
- `writing-for-agents` 把流程写成可复用纪律