把任务状态与数据目录分开，设计只读进度记录

完整操作教程 · 参考固定 · 往返恢复验收

导师需要掌握项目进度，不一定需要看到原始数据。把任务状态、里程碑和异常原因单独记录，可以在不暴露样本目录的前提下进行协作。

运行边界：示例状态不连接调度器；还需分别验证登录、API和静态文件的访问边界。

把“看进度”与“看数据”分成两种权限

项目负责人常常需要知道任务是否开始、完成了哪些阶段、有没有失败和预计何时能给出下一次更新；这并不必然要求读取FASTQ、临床表、单细胞对象或分析脚本。把进度页放在样本目录里，然后把整个目录只读开放，看似省事，实际可能暴露样本名、分组、工具参数、失败日志甚至个人信息。更清楚的做法是单独维护一个状态文件或任务看板，只写汇报所需的最小字段。

进度记录可以包括项目编号、任务编号、阶段名称、状态、开始和更新时间、负责人、阻塞原因类别、下一步、预计复核点和交付物版本号。不要在面向导师的状态字段中加入病人编号、样本文件路径、原始命令、访问令牌、数据库密码或详细研究假设。若阻塞原因需要技术诊断，可另建仅限执行团队访问的工单，状态页只显示“等待输入文件确认”或“软件环境冲突”等概括信息。

状态词应有明确定义。例如“待开始”表示输入和资源尚未核验；“运行中”表示进程实际存在或调度器确认已开始；“已完成待复核”表示命令退出成功且输出结构检查通过，但科学解释尚未完成；“已交付”需有版本号和交付记录。避免把脚本最后打印“done”直接当作完成状态。有些流程在中间步骤成功后会继续失败，进度应能表达部分完成和重跑。

状态字段	可展示示例	不宜放入只读进度页
阶段	原始数据质控、定量、统计复核	样本原始文件名
状态	运行中、待复核、阻塞	患者或受试者标识
更新信息	更新时间、责任人、下次检查点	密钥、完整命令和访问令牌
交付追踪	结果版本、文件摘要、提交日期	公开链接到敏感原始目录

状态从任务事件产生，不靠手工猜测

理想情况下，提交作业、开始执行、退出码、资源统计和输出验收由调度器或工作流系统产生；人工只补充原因解释、研究判断和下一步。没有工作流系统时，可以用简单的JSONL或CSV记录每个阶段的事件，但要统一字段、时区和任务ID。状态写入应先校验合法值，再使用临时文件原子替换，避免读者刚好打开到一半写入的文件。多人同时修改时应使用数据库、锁或单一写入者机制。

更新时间不能被自动刷新机制伪造。页面每分钟重新加载，并不说明分析任务有进展；可以将任务数据更新时间和页面访问时间分开展示。运行时间由调度器开始时间与当前时间推算，排队时间则另列，不能把等待资源算成程序运行。若状态服务不可用，显示最后更新时间和“状态暂未同步”，不要继续展示旧状态却标作当前。

示例：只生成不含数据路径的状态快照

下例从受控的阶段记录生成一份精简JSON。示例不连接任务调度器，也不读取任何原始数据，生产环境应由任务系统提供经授权的事件。执行人需校验任务ID和允许字段，文件权限只开放给执行团队和指定查看角色。若只读页面通过Web服务器发布，还要另外配置认证、缓存和访问日志策略。

```
import json, os, tempfile
from pathlib import Path
ALLOWED = {"queued", "running", "blocked", "review", "delivered", "failed"}
```

```
event = {
    "project_id": "P-DEMO", "task_id": "T-001",
    "stage": "count_matrix_qc", "status": "review",
    "updated_at": "2026-11-15T00:00:00+08:00",
    "owner": "analyst-01", "next_checkpoint": "confirm sample-level QC",
    "result_version": "v1", "data_path": None,
}
assert event["status"] in ALLOWED
assert event["data_path"] is None
out = Path("status-demo.json")
out.parent.mkdir(parents=True, exist_ok=True)
fd, tmp = tempfile.mkstemp(prefix=".status-", dir=out.parent)
try:
    with os.fdopen(fd, "w", encoding="utf-8") as w:
        json.dump(event, w, ensure_ascii=False, indent=2)
        w.write("\n"); w.flush(); os.fsync(w.fileno())
    os.replace(tmp, out)
finally:
    if os.path.exists(tmp): os.unlink(tmp)
```

这里的项目名和时间是合成示例。原子替换解决的是单个文件被读到半写入状态，不解决并发写冲突或身份鉴权。状态文件不能因为字段里不含原始路径，就自动视为可公开；项目名称、责任人和研究阶段也可能是敏感信息。发布前按接收者的最小知情范围筛字段，并确认外部访问需要登录、链接是否会被搜索引擎索引、缓存多久、谁能撤销访问。

设置只读角色，并验证“看不到数据”

如果进度页在内部系统中，优先创建独立的查看角色，只授权读取所需项目的状态接口；不要给服务器Shell账号后再口头约定“不要打开原始目录”。若导师通过共享文件查看，给单独的状态副本而不是样本目录的共享权限。测试账号应能读取指定项目进度，不能读取其他项目，不应能修改状态，不应能浏览原始数据目录。检查API返回、静态文件目录、错误日志、搜索索引和备份位置，避免只测网页首页。

导师如果点击“结果链接”，链接目标也需要单独设权限。可以先提供去标识化的汇总报告，让确有必要的人员申请更高权限。每次组内人员变化时复核账号和项目映射；离组成员应撤销状态页和文件访问两处授权。日志保存谁何时查看了哪一个项目，但避免记录敏感内容本身。

状态沟通仍由项目负责人主动推动

透明进度的目的是减少信息不对称，不是让客户通过频繁刷新页面来推动技术排期。内部负责人应主动设置检查点，遇到输入缺失、算力排队或结果异常时及时更新状态和下一步计划。页面可提供“最近更新时间”和“下次同步时间”，但不要用一个没有责任人的看板代替沟通。

如果团队希望建设多用户计算环境，先核对账号体系、目录隔离、项目角色和审批流程，再决定复用现有平台还是自建轻量状态服务。环境或服务器可以支持稳定的任务执行和权限管理，但具体用户隔离、数据托管、备份、审计和项目管理范围都需要按实际方案确认。

把阶段定义成可验证事件

为避免进度状态流于主观，给每个阶段指定进入条件、退出条件和证据。例如“数据接收完成”要求文件清单、大小和摘要检查通过；“质控完成”要求输出报告存在且结构检查通过；“统计分析待复核”表示程序退出码成功，但负责人尚未确认方法与结果；“已交付”要求版本号、文件清单和接收记录齐全。这样导师看到“完成”时知道它代表什么，分析人员也不会把程序成功误当成研究结论确认。

一个任务可能有多个并行步骤或失败重试。状态模型应允许部分完成、等待外部输入、暂停、重试中和已取消，并保留事件历史，不只覆盖当前状态。项目看板显示最新状态时，也应能查看更新时间和上次状态变更的简要说明。状态变更由任务管理系统或指定责任人写入；如果同一个任务被重复提交，使用独立attempt_id关联父任务，避免旧失败记录覆盖新成功状态。

进度估算应说明依据与不确定性。比如“预计周五复核”来自当前运行队列和剩余验证步骤，仍可能受上游输入补充、排队或软件异常影响。没有可靠估算时展示“待确认时间”，比给一个精确日期随后反复延期更可信。页面不要把运行百分

比当成真实进度，除非工作流能根据已完成样本或明确步骤计算。对耗时差异大的任务，可以展示已完成样本数和总数，但要确认样本数不会暴露敏感信息。

防止状态页成为第二个数据泄露面

只读状态也可能透露研究信息：项目名称可能暴露疾病方向，阶段名称可能透露候选靶点，报错内容可能包含服务器路径或样本ID，下一步计划可能包含尚未公开的研究设计。面向导师的状态页面应经过字段白名单筛选，不要直接把调度器日志、Notebook输出或分析错误堆栈嵌进去。对于外部合作方，可使用项目代号和阶段类别，具体细节通过已有保密沟通渠道确认。

网站缓存、浏览器历史、反向代理访问日志和静态文件链接都要纳入权限审计。一个需要登录的页面若同时把状态JSON放在可匿名访问的静态uploads目录，登录并没有保护数据。测试时用未登录请求、其他项目成员账号和授权账号分别访问页面及API；对每一类记录预期HTTP状态和返回字段。搜索引擎的robots规则不等于访问控制，随机文件名也不是可靠权限边界。

状态更新服务需要身份认证、项目授权校验和修改日志。只读角色不应能够修改状态，即使前端隐藏编辑按钮，也要在后端拒绝未授权写请求。对于人工录入的进度，保留修改人和更新时间；对于自动生成状态，记录数据源和同步错误。备份状态时考虑其敏感级别并设置访问范围，项目删除或成员退出后同步撤销看板、文件和API授权。

把协作节奏写在流程里

课题组可以约定固定同步频率，例如每个关键里程碑更新一次，而不是要求导师持续盯着任务页面。对阻塞状态，责任人应主动说明需要谁提供什么输入、影响哪一步、何时再评估；对长时间运行任务，可以只展示最近心跳时间和日志是否更新，不需要暴露命令全文。进度记录的目的，是帮助双方更早发现计划偏差和决策依赖，而不是将日常催办转移给客户。

任务结束后归档状态快照，保留哪些步骤完成、哪些未做、交付版本和未决问题。若任务结果后来被重跑，应新增运行记录并关联旧版本，而不是悄悄更换报告文件。项目结项后按合同和数据政策确定谁还能查看状态历史、保存多久以及如何销毁。简洁的状态看板只有在它和真实任务记录、责任人、数据权限及沟通机制一致时，才会带来透明度。

如果现有工作环境没有角色隔离能力，不要用“所有人共用账号”绕过限制。先确定查看者名单、项目映射和输出字段，再由管理员评估适合的访问控制方式；必要时用脱敏摘要替代在线系统。多用户计算环境可帮助统一任务执行和账号管理，但是否提供独立进度门户、审计日志或数据托管要以具体环境设计为准。

状态可视化要能指出谁在处理下一步

每个未完成状态都应有负责角色和下一检查点。若状态是“等待输入文件”，记录由谁提供、需要什么格式、预计何时复查；如果任务已失败，先判断是否已有安全的重跑条件，再决定是否更新为“重试中”。对于等待外部审批、软件许可或计算资源的任务，显示等待类型和责任方，避免一律标成“分析中”。这样导师不需要访问底层目录，也能知道是否需要自己做决策。

进度页面应有过期规则：运行状态多久没有心跳就变成“状态待核实”，已完成状态多久不更新就提示负责人复核。具体时限由任务类型决定，不要硬编码同一个小时数用于几十分钟任务和数天任务。页面展示当前时间时使用明确时区；更新时间从任务系统生成，避免浏览器本地时间造成误解。设置状态纠错入口，让负责人能指出状态与现实不符，并保留修订历史，而不是删除旧记录。

可视化可以简洁到一张里程碑表：阶段、当前状态、最近更新时间、负责人、下一检查点。颜色不能成为唯一信息，需同时写文本，方便打印和色觉差异用户阅读。不要为了让项目看上去顺利而隐藏失败步骤；显示“失败待排查”并附非敏感类别，比跳过该阶段更可信。对长周期分析，按里程碑而非每个shell命令展示进度，减少噪声。

配套练习包

本PDF对应的ZIP包含本文代码、主题模板或合成数据、README和SHA256校验文件。运行前核对包内清单，并按文章说明在独立目录验证。

参考资料

Linux 文件权限与 ACL 手册 · 来源资料

NIST 访问控制基础资料 · 来源资料

操作建议：先在合成小样本上验证流程，再依据项目参考、标签合同、存储策略与获配资源验收真实数据。