Bai 介绍完整版

社区协作助手「白」

摘要

社区协作助手「白」（Bai）是一个开源的社区协作平台，也是一个记录社区贡献的「贡献记录本」。它给每位社区成员一个 semi 就能登录的数字身份；社区在里头发任务、发公告、办活动，成员领任务、交凭证、拿积分。每一份为社区出的力都被写成一条可追溯的记录，公开、可信。它和社区数字身份 Semi 一起生长：Semi 认证「我们的身份」，Bai 记录「我们为社区做了什么」。既是记录当下的社区贡献激励模式，也是探索未来社区数字共治的可能性。

一、从一个社区说起

在我们的实地社区体验中发现，一个社区里，总有做不完的事：有人要帮厨做饭，有人要办一场活动，有人要修一段篱笆，有人要整理一堆文档，有人要带一期线上共学。这些事温暖又琐碎，靠的是大家自觉。一个社区能长期运转，靠的不是谁的声音大，而是每一件小事都能被看见、被记住、被回应。

可一件事只要放到线上，就难免走样：需求发在群里，刷两屏就沉了；报名靠接龙，谁报了谁没报要靠人点名；做完了，也只是在群里说一句「我弄好了」。社区越大、人越流动，这种「凭记忆办事」的方式就越吃力。人来人往，缺少共同的尺子。社区里有人住得久、有人只来两周，有人就在村里、有人远在千里之外。熟人社会里「我心里有数」就够用，人一多、人一换，就需要「规则公开、记录可查」。

社区不缺热情，缺的是一个让热情被看见、被留住的地方。

Bai 就是为了把这种日常协作搬到一个大家都能看见的地方而做的。

二、它是什么

社区协作助手「白」（Bai），是一个给社区用的协作与管理平台。

你可以把它想成社区的一间「公共大厅」：

- 墙上贴着公告，谁都不会错过社区正在发生的事；
- 桌上摆着任务墙，谁需要帮忙、谁能帮忙，都写在明处；
- 角落里有一本贡献记录本，记下每个人为社区做过什么。

它背后接入了社区数字身份 Semi——用同一个身份，成员可以在多个社区应用之间自由往来，不必到处注册账号；完成了贡献，记录会落回同一个身份名下。Bai 从社区的真实协作里长出来，但它不只属于某一个社区：任何村庄、合作社、社区或兴趣小组，都可以用它来组织自己的协作。

三、它现在能做什么

1. 用 semi 身份，并进入社区。 semi 登录后系统自动建好账户；通过 Semi 的数字身份，一个账号可以登录多个社区应用。第一次进入，可以从社区广场浏览、加入感兴趣的社区，之后随时在顶部切换。

2. 发布与认领任务——这是 Bai 最核心的功能。 发布者写清任务做什么、给多少积分、领取与提交的截止时间、需要提交什么样的凭证、是否需要多人协作；成员在任务广场里按标签、关键词、截止时间和积分排序来找活干，一键领取。任务可以限一人，也可以多人共同完成。

3. 提交凭证，审核闭环。 领取人上传做事的证明，发布者可「通过」「驳回重交」或「终止」。任务沿「未领取 → 已领取 → 待提交 → 审核中 → 已完成」流转，被驳回还能重新提交。全过程留在任务时间线上——谁，在什么时候，为社区，做成了什么。

4. 社区积分，让每一份付出都算数。 任务完成即发放社区积分。不同社区可以按自己的规则设立属于自己的积分，记录本地成员的贡献。积分不是奖金，而是一本大家共同的贡献记账本。

5. 办活动。 创建活动，写清时间、地点、人数与报名方式，开放报名并生成报名凭证；成员在活动日历里看到社区要发生什么，活动详情页完成报名。一场活动的发布、报名和参与者名单，都在同一处管理。

6. 社区商城与账单。 社区可以发布商品（含规格、库存与价格），成员加购、结算、付款。账单中心保留每一笔订单、每一项状态和链上凭证，支持继续付款、申请退款、查看收据与履约进度；付款依靠链上记录自动对账，不需要人工「确认收款」。

7. 公告与动态。 社区的公告、动态、成员和介绍集中展示在社区主页；管理员可以发布、置顶公告，维护社区的公共信息。

8. 消息与通知。 任务被领取、审核有结果、活动有动静、账单有变化，都会通过站内通知送达本人。

9. 个人主页。 头像、所属社区、社区积分、钱包地址与收款二维码，以及他发布和参与过的任务与动态，共同构成这个人在社区里的「故事线」。

四、我们为什么这样看待「积分」与「记录」

在社区里，积分不只是用来交易的，它更像一本大家共同的贡献记账本：有人煮的一锅饭，远方伙伴写的一行代码，都能在这本账上被公平地看见。它想做的，是把贡献从少数人的记忆，变成大家共同的记录；在社区，这份记录也是大家参与社区事务讨论、一起做决定时的重要依据。

我们在这套系统里特别看重「故事线」：一笔积分只说了一个数字，但配上任务、凭证、审核和时间线，它讲的是一件具体的事——谁，为社区，做了什么，什么时候做的，有没有做成。很多事情多年以后再翻看，这些记录依然在，不会因为换了负责人、换了系统就消失。

我们的初衷便是让所有人的贡献都能被看见，被记录；同时探索未来社区共治的可能性。记录当下，也是记录未来。

五、它是怎么做到的（技术篇）

这一节写给对技术感兴趣的朋友。不熟悉技术可以跳过。这些都能在 GitHub 上的公开仓库里找到。

1. 整体结构：三层分工

Bai 由三部分组成：

- 前端（mycoseed-frontend）：用 Nuxt 3 + Vue 3 写成的网页应用，Pinia 管理状态，Nuxt UI 和 Tailwind CSS 负责界面，部署在 Vercel。它负责展示和跳转，不持有任何密钥。
- 后端（mycoseed-backend）：Node.js + Express + TypeScript 的接口服务，数据存在 Supabase（PostgreSQL），图片和凭证也存在 Supabase Storage；部署在 Fly.io，附有 Dockerfile，可以搬到任何支持 Docker 的服务器上。
- 智能合约（contracts-project）：Solidity 编写、Hardhat 编译部署。目前上链的是备注合约 RemarkLogicV1，外面套了一层可升级代理（Proxy），以后修 bug 换实现、合约地址不变。

链上相关的逻辑集中在后端的 services/billing 目录里，前端不直接和链打交道。

2. 一项任务的一生：一次有据可查的状态流转

- 任务存在 tasks / task_info 两张表里：标题、说明、积分、领取截止、提交截止、需要什么凭证、是否允许多人、标签等。
- 状态沿「未领取 → 已领取 → 待提交 → 已提交 → 审核中 → 已完成」流转；被驳回可以重新提交，也可以终止。每一次变化都往任务的时间线里追加一条，记下状态、操作人和时间，不覆盖旧记录。
- 多人任务按人分别记录：谁领取、谁提交、谁通过，各自独立，互不混淆。
- 提交凭证时，后端先压缩图片，再存进 Supabase Storage，同时算一份 SHA-256 摘要留作核对。审核接口只对发布者开放，服务端每次都会核对操作者的身份。
- 状态一变，通知就写进 notifications 表，并按用户自己的通知设置决定发不发。前端从接口读取，不再靠人力查找。

让任务不是靠谁口头说「做完了」，而是每一步都有记录、有操作人、有时间。

3. 与 Semi 的配合：一套身份，完整生态

Bai 和 Semi 分工明确：Semi 是数字身份，Bai 是数字记录。两者通过统一登录和链上备注连接起来。

- 登录就是一次 OAuth 授权。 Bai 的后端作为 OAuth 2.0 客户端接入 Semi，采用「授权码 + PKCE」（即使授权码在传输中被截获，也无法换取登录凭证），并用 state 防止请求被伪造或混用。回调时用授权码换令牌、读取用户信息，再按 Semi 的用户编号找到或创建同一个人，同时记下他的智能账户地址。同一个 Semi 身份，在 Bai 里就是同一个用户；以后其他社区应用接入 Semi，也共用这一个身份，不必重新注册。
- 积分在 Semi 的智能账户里发。 每个 Semi 账户都是一个 Safe 智能账户（配合 ERC-4337 账户抽象标准），手续费由平台的代付服务承担。用户输入 6 位支付码，浏览器在本地解开密钥并签名。Bai 从头到尾不接触私钥，也不保存支付码。

- 从 Bai 跳转时带上「坐标」。 发布者在任务页点「发放积分」，Bai 打开 Semi 的转账页，并带上任务池编号、任务编号、一条默认备注（例如「任务：《标题》+ 时间」，可修改）和领取人写的那句感想。
- Semi 把两个编号用 keccak256 换算成链上的数字编号，调用备注合约的 saveRemark(poolId, taskId, senderRemark, receiverRemark)。「发积分」和「写备注」被放进同一个链上操作：要么都成功，要么都不发生。合约只存证、不碰钱，写完发出 RemarkSaved 事件。
- 记录能找回来，也能核对。 Bai 的后端用 Alchemy 按事件签名和编号哈希查链上日志，取回两句话和写入时间，与任务一一对应。前端把结果同步回来时，后端不会直接采信：它会读这笔操作的回执，确认发起地址确实是本人的智能账户、接收方也确实在操作里；成功与否、手续费多少都以链上为准。

这使身份从 Semi 来，贡献回到 Semi 去；中间的每次发放，Bai 都有记录，链上有存证。

4. 想自己部署

数据库迁移脚本都在 db/migrations 里，按编号顺序执行，附带回滚脚本；社区级开关可以按社区灰度启用，也可以一键回退。后端有 Dockerfile 和 Fly.io 配置；前端可以直接部署到 Vercel 或任何 Node 环境。链、代币、OAuth、短信等配置集中在环境变量里，缺项会在启动阶段直接报错。

想接入 Semi 的其他社区工具，只需在 Semi 里登记一个 OAuth 应用，配好回调地址和权限范围。

代码都在 GitHub 上（github.com/nantang-dao），欢迎查看、使用和参与改进。

六、下一步

Bai 现在的资金流走的是「点对点直付」的过渡实现：积分与付款通过 Semi 完成，依靠链上转账与备注自动对账。这是我们刻意先跑通的部分——结构完整、可以放心使用，但还不涉及托管与分账。

下一步，我们计划把托管合约正式部署上线，让社区里的钱真正「先托管、后放款」：

- 任务质押与托管。 发布者先把赏金锁定在合约里，验收通过才放款给领取人；未完成的部分按比例退回。
- 商城与活动的托管放款。 商城买家确认收货后放款，活动在可取消截止后放款给主办方；截止前可取消并退款。
- 平台费与社区费。 支持按类目、按社区设定费率，费用留在合约内、按月批量结算。
- 链上退款与对账。 放款、退款、费用计提都由合约事件驱动，对账不再依赖人工操作。

届时，社区里的每一笔付出，不仅有明确的故事线，也有可托管、可追溯、可结算的资金流。