task_bridge

محتوا

تخته وظایفتان و منبع حقیقت واحدتان، به‌طور بی‌نقص و دوسویه همگام‌سازی‌شده.

خلاصه

task_bridge در Go یک موتور همگام‌سازی عمومی، جداشده و دوسویه وظایف/تخته است. این ابزار منبع حقیقت واحد اقلام قابل‌انجام SQLite را با مستندات پیگیری‌کننده و تخته‌راه‌دور یک پروژه‌را با استفاده از منطق قطعی «آخرین ویرایش برنده (اولین هدف: کلیک‌آپ؛ جیرا/لینیر در برنامه) می‌شود»، اجرای آزمایشی پیش از اعمال و تضمین عدم خرابی داده‌ها همگام‌سازی می‌کند.

توضیح کوتاه

Go زیرماژولی مستقل از پروژه در SQLite (منبع حقیقت واحد) که اقلام قابل‌انجام را به‌طور دوسویه با مستندات پیگیری‌کننده و تخته‌راه‌دور (ابتدا کلیک‌آپ) همگام‌سازی می‌کند. منطق قطعی «آخرین ویرایش برنده می‌شود»، اجرای آزمایشی پیش از اعمال، وب‌هوک‌های تأییدشده با HMAC؛ هرگونه اعتبارنامه و شناسه در زمان اجرا توسط مصرف‌کننده تزریق می‌شود.

توضیح بلند

هر تیمی سرانجام دو دفتر ثبت کار یکسان‌را نگه می‌دارد: یکی واقعی — کد، مستندات، پایگاه داده داخلی — و دیگری که مدیران آن‌را می‌بینند، تخته‌ای مانند کلیک‌آپ. این دو به محض دستکاری هر یک از طرفین از هم فاصله می‌گیرند و تطبیق دستی آن‌ها دقیقاً همان کار خسته‌کننده و مستعد خطاست که هیچ‌کس به‌طور قابل‌اعتماد انجام نمی‌دهد. task_bridge برای از بین بردن این شکاف طراحی شده است؛ به این صورت که هر سه بازنمایی‌را به‌عنوان یک سیستم‌واحد در نظر می‌گیرد که باید همگام نگه داشته شوند: **منبع حقیقت واحد اقلام قابل‌انجام SQLite**، **یک پروژه مستندات**، اولین تخته پشتیبانی‌شده کلیک‌آپ است و جیرا و لینیر به‌عنوان — **تخته‌راه‌دور** و **پیگیری‌کننده** اعضای آینده برنامه‌ریزی شده‌اند. همگام‌سازی قطعی است (آخرین ویرایش برنده می‌شود)، ابتدا به‌صورت آزمایشی اجرا می‌شود و حول یک‌وعده غیرقابل‌مذاکره مهندسی شده است: هرگز داده‌ها‌را خراب یا گم نمی‌کند و هرگز به‌طور خاموش یک طرف‌را کهنه‌رها نمی‌سازد. در حوزه‌ای که یک همگام‌سازی بی‌دقت می‌تواند یک هفته کار‌را بازنویسی کند، این رویکرد ایمنی کل ماجراست. از نظر معماری، این یک زیرماژول سخت‌گیرانه است که توسط پروژه‌های دیگر مصرف می‌شود و کاملاً مستقل از پروژه است (بر اساس قرارداد جداسازی قانون اساسی §۱۱٫۴٫۲۸): هیچ مقدار خاص پروژه‌ای‌را ارسال نمی‌کند و هرگونه اعتبارنامه، شناسه تخته/پوشه، فیلد کلید آیتم و مسیر پایگاه داده در زمان اجرا از طریق `pkg/config.Config` توسط مصرف‌کننده تزریق می‌شود. این ماژول به‌صورت لایه‌ای تمیز طراحی شده است: یک CLI (`reconcile/push/pull/resolve/status/conflicts/`

و یک دیمن در حال اجرا (گیرنده وب‌هوک + تطبیق دوره‌ای)؛ یک کلاینت نازک بر روی `init` ؛ یک حل‌کننده که آدرس‌های تخته/پوشه‌را از طریق MIT با مجوز `raksul/go-clickup` ؛ یک نگاشت‌گر بین اقلام (URL بدون حدس زدن گزامر) به شناسه تبدیل می‌کند API کاوش‌های زنده قابل‌انجام محلی و فیلدهای وظیفه‌راه‌دور؛ یک موتور همگام‌سازی «آخرین ویرایش برنده می‌شود» با HMAC-SHA256 را با `X-Signature` نتایج صریح تعارض؛ و یک گیرنده وب‌هوک که امضای تأیید می‌کند. این ابزار در مورد بلوغ خود صادق است: این اسکلت اولیه اولویت اول است — ساختار، رابطها، نقاط ورود و مرز جداسازی در جای خود قرار دارند، اما منطق همگام‌سازی و فراخوانی‌های زنده کلیک‌آپ هنوز پیاده‌سازی نشده‌اند (هر استاب یک خطای صریح «پیاده‌سازی‌نشده» برمی‌گرداند، طبق قانون عدم استفاده از داده‌های جعلی).

چرا آن‌را ساختیم

تیم‌ها وضعیت «واقعی» کار‌را در کد/مستندات نگه می‌دارند، در حالی که مدیران در تخته‌ای مانند کلیک‌آپ زندگی می‌کنند — و این دو دائماً از هم فاصله می‌گیرند. `task_bridge` آن‌هارا به یک سیستم تبدیل می‌کند، به‌طور قطعی و ایمن همگام‌سازی می‌کند تا هیچ‌یک از طرفین کهنه یا نادرست نشود.

محتوا

چرا یک تغییردهندهٔ بازی است

همگام‌سازی دوسویه بوردها معمولاً یک یکپارچه‌سازی یک‌باره و سخت‌افزاری است که هر تیمی آن‌را به شکلی ناقص بازسازی می‌کند. `task_bridge` این مفهوم‌را به کتابخانه‌ای قابل‌استفادهٔ مجدد و تزریق‌شونده با اعتبارنامه‌ها بازتعریف می‌کند که تضمین‌های سخت‌گیرانهٔ ایمنی داده‌ها در آن تعبیه شده است — اجرای آزمایشی پیش از اعمال، تعیین اولویت ویرایش نهایی به‌صورت قطعی، و تأیید رویدادها با HMAC — تا هر پروژه بتواند با تزریق پیکربندی، به جای نوشتن یک اتصال‌دهندهٔ شکنندهٔ دیگر که به ساختار داخلی‌اش وابسته است، یکپارچه‌سازی قابل‌اعتماد بوردرا به کار گیرد.

نوآوری‌ها

- همگام‌سازی سه‌سویه دوسویه: SQLite (منبع واحد حقیقت) ↔ بورد از ↔ مستندات ردیاب راه دور
- جداسازی کامل (§۱۱،۴،۲۸): بدون مقادیر ثابت پروژه؛ همهٔ موارد در زمان اجرا تزریق می‌شوند.
- تبدیل زنده API به شناسه به جای تجزیه نحوی شکنندهٔ دستور URL.
- دریافت وب‌هوک با تأیید امضای HMAC-SHA256 برای رویدادهای زنده.

چالش‌ها و راه‌حل‌ها

- **ایمنی داده‌ها در سه منبع:** با تعیین اولویت ویرایش نهایی به‌صورت قطعی، اجرای آزمایشی پیش از اعمال، و نتایج صریح در مواجهه با تداخل حل شد.
- **قابلیت استفادهٔ مجدد بدون وابستگی:** از طریق مرز تزریق `pkg/config` (بدون ارسال جزئیات خاص پروژه) حل شد.
- **شناسایی قابل‌اعتماد بورد:** با تبدیل آدرس‌های اینترنتی به شناسه‌ها از طریق پروب‌های زنده API حل شد.

- با بازگرداندن خطاهای صریح «پیاده‌سازی‌نشده» به جای ارائه نسخه‌های جعلی **:داربست صادقانه** حل شد.

پشتهٔ فناوری (چرایی و چگونگی)

- **Go** — موتور، CLI (`cmd/task_bridge`) و دیمن (`cmd/task_bridged`).
- **SQLite** — منبع واحد حقیقت برای آیتم‌های قابل‌انجام.
- **`raksul/go-clickup` (MIT)** — بسته‌بندی انتقالی برای کلیک‌آپ.
- **HMAC-SHA256** — تأیید امضای وب‌هوک.
- **کرون + وب‌هوک‌ها** — تطبیق دیمن + دریافت رویدادهای زنده.
- **`pkg/config`** — مرز تزریق اعتبارنامه/شناسه در زمان اجرا.

> است — منطق همگام‌سازی هنوز پیاده‌سازی **داربست اولویت اول** صداقت وضعیت: این یک نشده است. آن‌را به‌عنوان محصول نهایی معرفی نکنید.