E06. Administration frontend analysis

Ilja Krasnjanski
24 October 2026 · Edited evidence report
Historical evidence through 24 July 2026; later engineering checks identified separately.

This report synthesizes the cited case records. Exhibits distinguish retained artifact excerpts from secondary analytical records and claims whose original inputs are unavailable. Excerpts may be redacted or reformatted; extraction methods and limitations are stated locally. Fingerprints identify inputs, not the truth of their claims. Bracketed S labels identify the analysis records listed at the end.

Acquisition record

The 23 July record describes recovery of seventy frontend modules by following imports exposed by a Vite development deployment. Embedded source maps supplied original TypeScript comments. The recorded acquisition did not use a login or invoke the authenticated data API. This report reviews the saved analysis, without contacting the deployment. [S1]

Interface observations

The client includes mail-status metrics, inbox and outbox views, templates, auto-reply controls, task boards and a database browser. A delivery-repository view uses clone URLs and availability labels with refresh controls. These components demonstrate an interface for managing messaging and monitoring repository availability; they do not supply verified sending volumes or suspension-response timings. [S1]

Wallet views expose import, balance preview, refresh, claim and bulk-claim operations. Mnemonic and ColdKey fields, together with Bittensor-related interface language, connect the intended workflow to collection targets found in the payloads. The source also refers to EVM and Aptos. No transaction receipt or backend execution log accompanies these interface observations.

Authorization and backend limits

A frontend helper accepts either a super-administrator flag or an email comparison. Its use in navigation and route guards establishes client behavior only. It does not prove backend enforcement or that a single account exclusively controls wallet operations. The personal mailbox value is omitted. Source comments suggest a Python/Django backend, but that backend was not recovered with the client. [S1]

The presence of a signup route does not demonstrate open registration. The record later identifies a disabled-signup message, qualifying its initial multi-operator interpretation. Neither a role model nor a task board establishes team size.

Assessment

The recovered interfaces connect delivery-repository monitoring and wallet-management functions in one application. This is evidence of software design and intended workflow, not proof of an end-to-end operational service or financial theft. The absence of wallet destinations in the reviewed frontend cannot establish that those destinations are unrecoverable elsewhere. Provider records, backend source, configuration or transactions would be required for stronger claims.

Exhibits

Exhibit E06-1. Admin API base and the backend route surface

Artifact: rms-platform admin client — `src/api/admin.tsx` (dev-mode module) · 19103 bytes · SHA-256 `eb3aa4f90ee3add3494d56e3c252ad4174080848f29946c8beee7f61b3907a1f`
Location: lines 13, 32, 90 (comments/constants)
Extraction: `sed -n '1,100p' src__api__admin.tsx`
Supports: E06 "Source comments suggest a Python/Django backend"; the delivery-repo and wallet route names.

```
export const ADMIN_API_BASE = import.meta.env.VITE_ADMIN_APP_API_HOST ?? "/api/";
// ...
 *      this interceptor was in place, the page would hit /api/github-stats/,
// ...
            // ROTATE_REFRESH_TOKENS=False in our SimpleJWT config means the server
```

The client points at a trailing-slash `/api/` base proxied to a Django backend (SimpleJWT auth); the backend itself was not recovered with the client. No redaction applied.

Exhibit E06-2. Delivery-repository monitor: clone_url and availability labels

Artifact: rms-platform admin client — `src/pages/StatsPage/GithubCheckPage.tsx` · 81876 bytes · SHA-256 `4267ff2fb31a951c9d0d5ea771c7e4ba98ff170918b31d2242898dd1836715a8`

Location: lines 15, 60-64, 89, 127-129, 154, 483

Extraction: `grep -n -E 'github-jobs-check|clone_url|children: "(Live|Down)"|freshnessTone|dead' GithubCheckPage.tsx`

Supports: E06 "A delivery-repository view uses clone URLs and availability labels with refresh controls."

```
// services.check_github_jobs every 5 min) instead of doing live provider HTTP,
            const { data } = await adminBase.get("github-jobs-check/", {
                    children: "Live"
                children: "Down"
    if (ageSec < 7 * 60) return "fresh";
    return "dead";
                                children: "clone_url"
```

The view polls `github-jobs-check/` and renders each delivery repo's `clone_url` with Live/Down and fresh/dead freshness tones. Confirms the `/api/github-jobs-check/` route and the `clone_url` field. No redaction applied.

Exhibit E06-3. Mail-status metrics and auto-reply

Artifact: rms-platform admin client — `src/components/ecommerce/EmailStatusMetrics.tsx` · 8742 bytes · SHA-256 `741ab0364fa1fbccdc8ca924d581b96992d03153c61d944b6d41011a5329edac`

Location: lines 30-65 (metric card titles)

Extraction: `grep -n 'title:' EmailStatusMetrics.tsx`

Supports: E06 "The client includes mail-status metrics ... auto-reply controls".

```
                title: "Total sent",
                title: "Sent today",
                title: "Replies today",
                title: "Auto-replies today",
```

A send-and-reply dashboard; `db/tables/email_inbox/auto-reply/` elsewhere in the client renders the automated replies. No redaction applied.

Exhibit E06-4. Wallet console: the four /api/admin/wallets/ operations and "Claim All"

Artifact: rms-platform admin client — `src/pages/Admin/WalletBalancesPage.tsx` · 109637 bytes · SHA-256 `f9b1b0c3e259d90ca35e827dc84cb137335b0a79420c532d4ffa0988bd994954`

Location: lines 70, 449, 478, 492, 512, 544-545, 639

Extraction: `grep -n -E 'admin/wallets/|Claim All|SuperAdmin coldkey|\.env' WalletBalancesPage.tsx`

Supports: E06 "Wallet views expose import, balance preview, refresh, claim and bulk-claim operations."

```
                const { data } = await adminBase.post("admin/wallets/preview/", {
            const { data } = await adminBase.get("admin/wallets/", {
            const { data } = await adminBase.post("admin/wallets/refresh/");
            const { data } = await adminBase.post("admin/wallets/claim/", {
            const { data } = await adminBase.post("admin/wallets/claim-all/");
                title: "ColdKey",
                description: "SuperAdmin coldkey balances"
                                    children: claimingAll ? "Claiming…" : "Claim All"
```

The page issues `preview/`, `refresh/`, `claim/` and `claim-all/` POSTs under `admin/wallets/`; the bulk button is labelled "Claim All" and its confirm dialog reads "Send all wallet balances to the claim recipients in .env?". ColdKey/mnemonic import language ties to the Bittensor targeting. No redaction applied (no addresses or secrets present in the client).

Exhibit E06-5. SuperAdmin helper logic (compared email REDACTED)

Artifact: rms-platform admin client — `src/utils/superAdmin.ts` · 1574 bytes · SHA-256 `3a6abe7822d0807a722db687f4f0557e9fdde89f4804761b80d4b44bc8ac2116`

Location: full module (lines 1-6), plus guard use in `src/routes/SuperAdminRoute.tsx` (SHA-256 `639d6bcf1507d6e9a9820dca2562b097a88bae24c8f8750d345214cf66a46512`)

Extraction: `sed -n '1,6p' src__utils__superAdmin.ts`

Supports: E06 "A frontend helper accepts either a super-administrator flag or an email comparison ... The personal mailbox value is omitted."

```
export const SUPER_ADMIN_EMAIL = "[REDACTED: operator email / personal-data lead]";
export function isSuperAdmin(user) {
    if (!user) return false;
    if (user.is_super_admin === true) return true;
    return (user.email || "").trim().toLowerCase() === SUPER_ADMIN_EMAIL.toLowerCase();
}
```

The helper returns true on either `is_super_admin === true` or a case-insensitive match of the logged-in email against a hardcoded constant; `SuperAdminRoute.tsx` calls `isSuperAdmin()` to guard `/admin/wallets`, else `<Navigate to="/stats">`. This is client-side route-guard behaviour only. The hardcoded email value is redacted (operator-identity lead, per brief).

Exhibit E06-6. Signup route exists but registration is disabled

Artifact: rms-platform admin client — `src/components/auth/SignUpForm.tsx` · 62125 bytes · SHA-256 `c39f4cbb829d0aa623a3d93bcb5798665406d72b4d1cb17552faa6e224b998ba`
Location: lines 16, 32, 44, 47
Extraction: `sed -n '10,50p' src__components__auth__SignUpForm.tsx`
Supports: E06 "The record later identifies a disabled-signup message, qualifying its initial multi-operator interpretation."

```
const SIGNUP_DISABLED_ALERT = "You can't sign up now.";
// ...
        window.alert(SIGNUP_DISABLED_ALERT);
// ...
        window.alert(SIGNUP_DISABLED_ALERT);
    const formDisabled = true;
```

The `/signup` route renders but the form is hard-disabled and alerts on mount and submit, so the route's presence does not demonstrate open registration. No redaction applied.

Claims without a retained primary artifact

- The Python/Django backend (`apps/admin/server`, `views.py`, `logs/check_github_jobs.log`): only referenced in client comments/strings; the backend source is not in the workspace.
- Verified sending volumes, suspension-response timings, wallet destinations, transaction receipts and backend execution logs: the report explicitly notes their absence, and none are retained.
- The role model / task board establishing team size: task-board modules exist but carry no retained data proving team size (consistent with the report's caution).

Exhibit inputs

Retained inputs identified by size and SHA-256. These include artifacts and analytical records; each exhibit identifies its evidence type and extraction limitations. The files themselves are not distributed.

rms-platform admin client — src/api/admin.tsx (dev-mode module) · 19,103 bytes
`SHA-256 eb3aa4f90ee3add3494d56e3c252ad4174080848f29946c8beee7f61b3907a1f`

rms-platform admin client — src/pages/StatsPage/GithubCheckPage.tsx · 81,876 bytes
`SHA-256 4267ff2fb31a951c9d0d5ea771c7e4ba98ff170918b31d2242898dd1836715a8`

rms-platform admin client — src/components/ecommerce/EmailStatusMetrics.tsx · 8,742 bytes
`SHA-256 741ab0364fa1fbccdc8ca924d581b96992d03153c61d944b6d41011a5329edac`

rms-platform admin client — src/pages/Admin/WalletBalancesPage.tsx · 109,637 bytes
`SHA-256 f9b1b0c3e259d90ca35e827dc84cb137335b0a79420c532d4ffa0988bd994954`

rms-platform admin client — src/utils/superAdmin.ts · 1,574 bytes
`SHA-256 3a6abe7822d0807a722db687f4f0557e9fdde89f4804761b80d4b44bc8ac2116`

rms-platform admin client — src/routes/SuperAdminRoute.tsx · 3,674 bytes
`SHA-256 639d6bcf1507d6e9a9820dca2562b097a88bae24c8f8750d345214cf66a46512`

rms-platform admin client — src/components/auth/SignUpForm.tsx · 62,125 bytes
`SHA-256 c39f4cbb829d0aa623a3d93bcb5798665406d72b4d1cb17552faa6e224b998ba`

Source records and fingerprints

SHA-256 identifies the reviewed source bytes, not acquisition time or every artifact described. Restricted originals are not reproduced.

[S1] Administration frontend recovery, 23 July 2026 · 8,864 bytes
`SHA-256 ea8e598980c0fefac94efbae41b356fa408d26df6cd890be7149f22289e60b9c`