Architectural Review

Architectural Review — ternent-api (PixPax/Stickerbook)

Generated: 2026-02-24

Executive summary
The codebase is currently a single Express application that mixes:
- platform auth/session logic (better-auth),
- account/user management,
- PixPax/Stickerbook domain logic,
- persistence adapters (Postgres + Spaces + local JSON),
- admin/billing utilities,
- and some server-side rendered pages.

This tight coupling increases regression risk and contributes to instability.

High-impact findings (actionable)

1) Avoidable hot-path I/O
- `main.mjs` reads `package.json` for every request to set `x-api-version`.
  - **Fix:** read once at startup and cache.

2) Inconsistent error handling
- Many routes use ad-hoc try/catch with inconsistent status codes (often 400 for all failures).
- No single error middleware enforcing a uniform envelope.
  - **Fix:** introduce a typed error model + global error handler.

3) Security gaps
- `/my-costs` triggers DigitalOcean billing API calls without robust authz.
- WebSocket channel accepts unauthenticated connections and parses JSON without guards.
  - **Fix:** require auth handshake; validate payload size/schema; lock down billing.

4) Ledger persistence is not concurrency-safe
- Pixbook ledger stored as a single JSON object in Spaces with unconditional overwrites.
  - Two devices can overwrite each other, causing data loss.
  - **Fix:** switch to append-only receipts/events (your chosen model) + snapshots.

5) Domain logic embedded in routes
- Large route modules (notably `routes/account/index.mjs`) contain transport + domain + persistence.
  - **Fix:** move to module services + repositories + adapters; keep routes thin.

6) Observability gaps
- Mostly console logging; no request IDs; no consistent structured context.
  - **Fix:** pino + request-id middleware + basic metrics.

Recommended target: modular monolith + Concord-style receipts
See `specs/SPEC-1.md` and `diagrams/*.puml`.

Key moves:
1. **Spaces becomes canonical receipt store** (append-only).
2. **Vault Transit signs receipts** (ECDSA P-256, custodial identities).
3. **Postgres becomes projection** (rebuildable).
4. **Explicit commands** produce receipts; queries read projections/snapshots.

Proposed refactor roadmap (minimize risk)

- Phase 1: stability/security patch (no functional change)
- Phase 2: modular extraction + validation + consistent contracts
- Phase 3: receipt streams + signing + verification
- Phase 4: projector/replay tooling + snapshots