Smart Home Energy Monitor ? Improvement Ideas

1. Modularize EnergyProvider \(src/contexts/EnergyContext.tsx:1-660\)
- Why: The provider owns connection management, persistence, automation rules, alerts, budget logic, and device CRUD inside one 600+ line component. Splitting it into focused hooks \(e.g., useDeviceStore, useAutomationRunner, useBudgetAlerts\) would reduce re-render surface area, simplify testing, and allow lazy loading of heavy logic when specific screens need it.

2. Stabilize localStorage-driven effects \(src/contexts/EnergyContext.tsx:114-141\)
- Why: Several effects list ‘localStorage.getItem\("onboarded"\)‘ and ‘localStorage.getItem\("brokerUrl"\)‘ as dependencies, creating new string references every render and causing reconnect attempts even when nothing changed. Reading the values once into state \(or subscribing to the storage event\) makes the dependency arrays stable and prevents unnecessary MQTT reconnect loops.

3. Validate broker changes before persisting \(src/pages/Settings.tsx:63-81,) Tj 0 -14 Td (src/pages/Or
- Why: Settings currently saves the new broker URL and shows a success toast before checking whether ‘mqttService.connect‘ actually succeeds. When the URL is wrong, the app stays offline but keeps the bad value in storage. Reusing ‘connectAndWait‘ with proper error handling \(and only persisting on success\) keeps users from locking themselves out and gives clear feedback.

4. Debounce budget alert toasts \(src/contexts/EnergyContext.tsx:612-646\)
- Why: Once the daily ratio crosses 75/90/100% the interval continues to send the same toast every minute, quickly overwhelming the UI. Tracking the last emitted level and only firing when the user crosses into a new band \(or when the date changes\) keeps alerts informative instead of noisy.

5. Consolidate auth token strategy \(Backend/middleware/auth.cjs:1-124 vs) Tj 0 -14 Td (Backend/utils
- Why: The backend ships both a custom HMAC token implementation and a jsonwebtoken-based helper that is never used. Maintaining two code paths invites drift and security regressions. Pick one approach \(ideally the proven library\) and delete the unused module to simplify audits and future hardening.