RemoteIQ — Frontend Master Specification (Gemini-Focused)

Objective: Deliver a modern, secure, technician-optimized RMM UI that is responsive, accessible, and  
aligned to the one-feature-at-a-time roadmap. This specification defines the complete Information Architecture,  
navigation, routes, components, state management, data contracts (from a frontend perspective), and testing strategy.  
Stack decision: Next.js 15 (App Router) + TypeScript + Tailwind + shadcn/ui + TanStack Query + Zustand + Zod + Playwright + Vitest + OpenAPI-driven clients.

# 1) Design Principles

• Technician speed first (fewest clicks to common actions).   
• Progressive disclosure (endpoint page contains deep tools; sidebar is for module entry).  
• Predictable IA with tenant scoping: Organization → Location → Department → Agent Type → User.  
• High contrast, keyboard- and screen-reader-friendly (WCAG AA).   
• Deterministic error handling (retry, report, resolve).   
• Feature-flagged rollout tied to phases.

# 2) Global Layout & Theming

# Phase F1 — Frontend Design & Layout — Dashboard UI Design Prompt (Gemini)

Design a modern dashboard UI for a professional IT software called RemoteIQ — a next-generation Remote Monitoring and Management (RMM) platform for MSPs.

Inspiration: Tactical RMM layout (left client/site tree, right data table) but significantly more modern, elegant, and high‑tech.

Core Layout Requirements:

• Left sidebar (vertical navigation) showing a client tree with collapsible organizations and sites.

• Main content area with a data table showing endpoint info:

– Device type icons (Windows, macOS, Linux, etc.)

– Status indicators (online, offline, pending)

– Columns: Client, Site, Hostname, Description, User, Security/Verified, Last Response, Boot Time, Power, Signal/Latency.

• Toolbar/top bar with search, notifications, settings, and account controls.

• Lower sub-navigation for: Summary, Checks, Tasks, Patches, Software, History, Notes, Assets, Debug, Audit.

Style & Design Goals:

• Dark mode first, plus a light variant.

• Flat, minimal components with subtle depth (soft shadows) and rounded corners.

• Clean typography (Inter/Roboto/Poppins).

• Accents: green=healthy, amber=warning, red=critical, blue=neutral info.

• Scales from 1080p to ultrawide; sticky headers; responsive tables; compact density toggle.

Optional Enhancements:

• Top insight bar for key metrics (CPU, RAM, uptime, agent version, alert count).

• Subtle hover/active transitions; focus rings (accessible).

• Column visibility manager; Saved Views; export (CSV/JSON).

Output Request:

• High‑fidelity UI mockups (not wireframes).

• Two variants: (1) Main dashboard overview; (2) Detailed device view (tabs for software, checks, history, tasks, patches).

• Provide state examples (empty, loading, error, filtered).

Mood: Professional, intelligent, futuristic — a premium IT automation platform.

• Left Sidebar (260px; collapsible to 72px), sticky.   
• Content Area with breadcrumb: /Organization / Location / Section / Item.   
• Top bar: search, notifications, user menu, theme toggle, help.   
• Theme: dark/light with tenant branding (logo, accent color).   
• Motion: subtle, respects prefers-reduced-motion.

# 3) Sidebar Navigation (Authoritative)

Primary items (with children):  
- Dashboard  
- Organizations  
- Locations  
- Departments  
- Devices  
- Automation: Run Script, Script Library, Schedules, Runs, Onboarding Playbooks  
- Tools: Remote Desktop, Remote Shell, File Browser, Services/Processes, Event Logs  
- Patching  
- Software  
- Reporting  
- Settings: Users & Roles, Policies, Audit Log  
- Help & Docs  
Behavior: accordion expansion, active highlight (border-left), tooltips in collapsed mode, audit events on navigation.

# 4) Route Map (Next.js App Router)

/dashboard  
/organizations  
/organizations/[orgId]  
/organizations/[orgId]/locations  
/organizations/[orgId]/locations/[locId]  
/organizations/[orgId]/locations/[locId]/endpoints  
/devices  
/devices/[deviceId] (endpoint page with all tools as tabs)  
/automation/run (wizard default)  
/automation/scripts  
/automation/schedules  
/automation/runs  
/automation/onboarding  
/tools/remote  
/tools/shell  
/tools/files  
/tools/services  
/tools/events  
/patch  
/software  
/reports  
/settings/users  
/settings/policies  
/settings/audit  
/help

# 5) Endpoint Page (All tools live here)

Tabs: Overview, Remote, Checks & Alerts, Patch, Software, Scripts, Inventory, Files, Services, Event Logs, Reports, Timeline, Notes.  
Quick Actions bar: Start Remote, Run Script, Patch Now, Reboot/Shutdown/Start, Tag/Move Department.  
Context Drawer: parameters for runs, live logs, artifact links. Bulk actions only appear in Devices list, not here.

# 6) Organizations → Locations → Departments → Agent Type → User

Filters on Devices list support multi-select for Location, Department, Agent Type, Status, Tags, OS, Last Seen.  
Saved Views capture filter JSON and column config; URL persists state; keyboard shortcut Ctrl/Cmd+K pre-scopes search to current Organization/Location.

# 7) Automation (Main Navbar) — Run Script Wizard

Step 1: Target selection — specific endpoints, filter-based (Organization/Location/Department/Agent Type/Tags/Status/OS/Last-Seen), or Saved View.  
Step 2: Script selection — search, language, tags, version pin, compatibility badges, provenance.  
Step 3: Parameters & Secrets — Zod-typed params; secrets referenced by key and injected server-side; optional dry-run.  
Step 4: Run now or Schedule — recurring (CRON/UI), throttle/jitter, maintenance windows, approval gates if policy requires.  
Live Run Drawer: per-device progress, output streaming, artifact links, CSV/JSON export.  
Performance: resolve count for 50k filtered endpoints < 1.5s p95; enqueue 10k-device run < 5s.

# 8) Components & State

Key components: <SideNav/>, <TopBar/>, <DeviceTable/>, <EndpointTabs/>, <TargetSelector/>, <ScriptCard/>, <RunScriptWizard/>, <LiveRunDrawer/>, <FiltersRail/>, <SavedViews/>.  
State: TanStack Query for server cache; Zustand for UI state (drawer open, selections); optimistic updates only on idempotent ops.  
OpenAPI client generation to ensure type-safe API contracts; Zod mirrors server constraints.

# 9) Accessibility & i18n

All interactive controls have ARIA roles/labels, focus rings, and keyboard shortcuts. Tables support screen readers (aria-describedby for pagination, sorting). All copy in i18n resources.

# 10) Real-time & Offline

WebSocket channels for device status updates, job streaming, alert feed. Graceful reconnection with exponential backoff. Offline queues for UI actions with clear conflict resolution.

# 11) Error/Empty/Loading States

Skeletons for lists and detail; Error boundaries per section; clear recovery actions (retry, open logs); Empty-state CTAs (upload first script, create first playbook).

# 12) Testing Strategy (Frontend)

Playwright e2e: navigation, wizard flows, device actions, accessibility checks. Vitest unit tests: hooks, reducers, render guards. Visual regression for critical pages. Performance budgets (LCP < 2.5s, TTI < 3s typical).

# 13) Security (Frontend Concerns)

Strict Content Security Policy; secure cookies; no secrets in client; mTLS only at agent/server; JWT short TTL with silent refresh; CSRF safe methods only; sanitize any rendered script outputs.