Telco ADB Automation – v1.0.0 Functional Scope

*This document inventories everything that is **present and working** in the first production release (v1.0.0). Use it as the reference when demoing the product, validating QA scope, or creating downstream documentation.*

---

1. Product Overview

- **Platform**: Electron desktop shell that boots a React 18 renderer and a FastAPI backend.
- **Primary value**: Reliable Android device management over ADB, sequential module execution (single-run + workflows), and live visibility on device state.
- **Distribution targets**: Windows 10/11 (NSIS installer, MSI for enterprise rollout, portable `.exe`).
- **Documentation**: Architecture, backend, and frontend guides authored in Markdown with PDF/DOCX exports (`docs/exports/`).

---

2. Desktop Shell & Packaging

Capability	Details
Electron main process	Launches FastAPI (`simple-server.py`), forwards stdout/stderr to renderer console, enforces preload isolation.
Auto-update	Configured to pull releases from GitHub; update check wired in main process.
Installers	NSIS wizard, MSI (silent install / GPO), portable build. All create Start Menu + desktop shortcuts and optional protocol handlers.
Localization	Electron menu + renderer copy available in English and French (`src/electron/locales`).
System tray	Running indicator and quick actions (show/hide, quit).

---

3. Backend (FastAPI)

Area	Implemented endpoints / services
Device API	`GET /api/v1/devices`, `GET /api/v1/devices/{id}`, `POST /api/v1/devices/{id}/alias`, `POST /api/v1/devices/{id}/setup`, stats endpoints.
Modules API	`POST /api/v1/modules/{module_id}/execute` for single-run; workflow runner uses same endpoint sequentially.
Workflows API	`POST /api/v1/workflows/run`, `POST /api/v1/workflows/stop`, plus helpers for draft storage.
Health	`GET /health` for Electron readiness checks.
Logging & storage	Structured logger (JSON lines), SQLite via SQLAlchemy for persistent metadata, JSON caches for device history.
Telco modules	`modules/telco_modules.py` implements ping, call, airplane mode, etc. Ping runs entirely server side; status returned to frontend without popup.

Operational behavior:

- Each module call is synchronous; response returned only when the ADB script finishes.
- Workflow runner waits for module n to resolve for all targeted devices before kicking off module *n+1*.
- Stop workflow sets cancellation flags that abort the next module dispatch loop.

---

4. Frontend (React 18 + MUI)

Page	Functional elements present today
Dashboard	Device table/grid with live connection state, alias editing inline, quick module shortcuts, start workflow CTA. Device cell shows `DeviceName Alias AliasValue` formatting requested by product.
Device Manager	Lists every device ever seen, including disconnected ones with cached metadata, history icon (log notebook) opens per-device timeline with timestamps of modules/workflows executed.
Modules	Card list of telco modules; each has RUN button (Ping button turns green during backend execution). Modules that require parameters surface forms.
Workflows	Builder UI with left/right panels, drag-and-drop from module palette, start/stop controls, per-module progress indicator (now blocks until completion).
Reports	Placeholder cards for upcoming analytics plus access to exported docs.

Shared UI behaviors:

- Device connect/disconnect polling interval tightened so UI reflects real hardware changes within seconds.
- When devices disconnect they are hidden from "active" list faster to avoid zombie cards.
- Alias edit icon aligned to the right for readability; "Alias" label text removed per UX feedback.

---

5. Device Lifecycle Features

1. **Discovery**: Automatic USB enumeration via `adb devices`, debounced to avoid flicker.
2. **Metadata**: For each device we capture serial, model, manufacturer, Android/API version, battery %, charging state, network operator, radio access technology.

3. **Alias**: Editable per-device alias stored server-side; inline text field in Dashboard (edit icon repositioned).
4. **History log**: Device Manager caches executed modules/workflows with timestamps; notebook icon opens the history modal.
5. **Speed optimizations**: Reduced debounce intervals for both connection and disconnection, so phone cards appear/disappear quicker.

---

6. Module & Workflow Execution

Standalone modules

- Ping, Call, Airplane Mode, etc. can be triggered individually.
- Ping runs entirely in FastAPI; RUN button colors: default → green (running) → default again on completion, with no toast message.
- Failure states surface in activity logs; e.g., Ping failure text displayed without blocking dialogs.

Workflows

- Workflow definitions stored in frontend state/local storage; Modules page supplies drag sources.
- Execution loop ensures module order is respected even for multi-step modules (Call waits for both dial attempts).
- UI progress indicator stays on current module until backend confirms completion (fix for premature highlight on module *n+1*).
- Stop Workflow button signals backend to halt; frontend stops advancing progress once cancellation acknowledged.

---

7. Device Manager Enhancements

Implemented requests from the product backlog:

- **Drag-and-drop** between module list/workflow canvas operates left right.
- **Pinned/Collections/Usage widget removal**: Legacy cards removed for cleaner layout.
- **History icon** with "notebook" glyph showing summary of actions + timestamps.
- **Advanced concepts (filters, timelines, bulk actions, notes, reports)** drafted in design docs; Device Manager currently exposes basic filtering + history but not the full advanced suite (documented as roadmap).

---

8. Documentation Assets

- `docs/architecture-overview.md`, `docs/backend-guide.md`, `docs/frontend-guide.md` plus DocFX `toc.yml`.
- CLI export pipeline: `docker run --rm -v "$PWD:/data" pandoc/latex <doc>.md -o docs/exports/<doc>.pdf` (+ `.docx`). Generated artifacts live under `docs/exports/`.
- Supplementary scripts (`generate_pdf.py`, `generate_release_pdf.bat`) to automate bundling.

---

9. Known Limitations (v1.0.0 truth set)

Area	Status
Background scheduler / cron	Not implemented yet (documented in roadmap).
Live ADB log streaming	Pending – WebSocket channel not wired.
Alerts/notifications (email/Slack)	Not built.
API authentication / RBAC	No auth in this version.
Advanced analytics / PDF workflow reports	Only device activity log exists.
CLI/API for CI pipelines	Future work.

These limits are captured in `RELEASE_NOTES_v1.0.0_ACTUAL.md` and should be reiterated to stakeholders when positioning the release.

---

10. Validation Summary

- **Manual QA**: Verified alias editing, ping run visual indicator, sequential workflow progression, faster device appearance/disappearance, removal of deprecated Device Manager widgets.
- **Backend smoke**: `simple-server.py` logs show server startup success; FastAPI endpoints reachable via Electron dev harness.
- **Doc exports**: Architecture, backend, frontend guides exported to PDF + DOCX; release notes exported as well (warning about emoji glyphs noted).

Prepared for v1.0.0 release to provide an exact inventory of what the application offers today.