**01 – Vision Brief (Technijian SDLC)**

**Project**

**MCPX-KendoBridge Admin Portal** — a secure, Azure SSO–gated admin UI and API that monitors, configures, and operates Technijian MCP servers and related services.

**Problem & Why Now**

Technijian teams need a single, secure control plane to:

* View health/readiness of MCP services and dependencies
* Manage configuration via DB-backed AppConfig/FeatureFlag/Lookup (no hard-coding)
* Initiate admin/ops workflows and observe streaming task progress (SSE)
* Enforce consistent environments and evidence for audits (HIPAA/FFIEC/PCI where applicable)

Fragmented tools, ad-hoc scripts, and prototype UIs slow delivery, complicate audits, and risk misconfigurations. A standardized portal + API aligned to our SDLC resolves this.

**Goals (What “Good” Looks Like)**

1. **Secure access**: Azure AD (Entra ID) SSO with app roles/AAD groups for admin access.
2. **Operational clarity**: Live health/ready signals, streaming job output, error taxonomy.
3. **Policy by default**: DB-backed configuration and feature flags; add-only schema; SP-only data access; no hard-coded settings.
4. **Repeatable delivery**: GitHub-first pipelines, four environments (Alpha → Beta → RTM → Prod) with quality gates and evidence pack.
5. **Design → Code fidelity**: Figma Make prototype mapped 1:1 to **KendoReact (Fluent 2)** components via ThemeBuilder token export.

**Non-Goals**

* Public/anonymous access (admin-only).
* Replacing existing vendor consoles (we integrate and orchestrate; we don’t duplicate their UIs wholesale).
* Building a generalized BI/reporting warehouse (basic exports OK; heavy analytics is out-of-scope).

**Key Outcomes & Success Metrics**

* **Access control**: 100% of admin interactions require AAD sign-in; least-privilege app roles enforced.
* **Reliability**: p95 API latency ≤ 400 ms JSON; streaming TTFB ≤ 200 ms; 30-day error budget respected.
* **Change safety**: 100% config changes audited; 0 prod secrets in repo; CI gates (build/tests, CodeQL, Dependency Review, Secret Scanning) pass on every merge.
* **Design fidelity**: All Figma frames map to shipped Kendo components; WCAG 2.2 AA passes axe checks in CI.
* **Audit readiness**: Evidence pack attached to each release and retained ≥ 1 year.

**Stakeholders & RACI (condensed)**

* **Accountable**: Director of Software Engineering (DoSE)
* **Responsible (docs)**: Doc-Factory role (this project’s documentation owner)
* **Consulted**: Systems Architect, Security/Compliance, DBA, Dev Leads
* **Informed**: QA, Client Services, MSP Ops

**Scope (Phase 1–2)**

**Admin UI**

* Azure SSO login → Dashboard (health/ready, version, uptime, queue depth)
* Configuration pages: AppConfig/FeatureFlag/Lookup (read/write via SPs)
* User access: assign app roles or group membership (AAD)
* Jobs/Tasks: launch, stream status via SSE, error envelope surfaced
* Audit trail: who changed what, when (config/actions)

**Admin API**

* JSON default; **SSE** when Accept: text/event-stream
* Endpoints for /healthz, /ready, /config/effective, /config/\* mutations (SP-backed), /jobs/\* (start/stream/result), /users/\* (AAD lookup read-only)
* Error envelope and rate-limit policy (include 429)

**DevEx/CI/CD**

* GitHub Actions: CI (build/test/coverage, lint, axe a11y, CodeQL, Dependency Review, Secret Scanning), Deploy (Alpha → Beta → RTM → Prod) with environment protections
* Evidence pack generation and retention (≥ 1 year)

**Out of Scope (initial)**

* End-user (non-admin) experiences
* Multi-tenant billing/quotas (track as future epic)

**Assumptions**

* Azure AD tenant available; app registration with redirect URIs provided
* SQL Server 2022 reachable from app tiers; managed identity or secretless pattern preferred
* Technijian network controls (Meraki/Cloudflare) in place; CORS origin allow-list managed per env

**Constraints (Tech & Policy)**

* **Tech stack**: .NET 8 (API/Workers), SQL Server 2022, React + **KendoReact (Fluent 2 theme)**, GitHub Actions, Figma Make + ThemeBuilder
* **DB rules**: *Add-only schema*. *Stored-procedure-only* access from app/API. *NoHardCoding*: all config/flags/lookups come from DB SPs or a secret store; never from code constants.
* **Environments**: **Alpha → Beta → RTM → Prod**. RTM validates against Prod DB (read-only) to ensure parity before Prod.
* **Security**: Azure SSO (PKCE), app roles/groups, CORS allow-list, centralized error taxonomy, rate limiting, audit logging.
* **Accessibility**: WCAG 2.2 AA. Axe checks run in CI.

**High-Level Architecture (text)**

* **Web (React/Kendo)** ↔ **Admin API (.NET 8)** ↔ **DB (SQL Server)**
* **Azure SSO** issues tokens; API validates JWT (bearer).
* **Config/Flags/Lookups** only via SPs (sp\_Config\_GetValue, sp\_Feature\_IsEnabled, sp\_Lookup\_Get, etc.).
* **Jobs** emit **SSE** events with heartbeats; UI shows live progress.
* **Observability**: structured logs with requestId, metrics for p50/p95/TTFB, uptime, queue depth; alerts on SLO breaches.

**Design → Code SOP (Figma → Kendo Fluent 2)**

1. Prototype in **Figma Make**. Name frames by route (/login, /dashboard, /config, etc.).
2. Export tokens to **ThemeBuilder** and generate Fluent 2 overrides.
3. Implement UI with **@progress/kendo-react-**\* and **@progress/kendo-theme-fluent**, applying ThemeBuilder tokens.
4. Replace any non-Kendo widgets from the prototype with the nearest **KendoReact Fluent 2** components (Grid, Charts, Dialogs, Inputs, Form, Drawer/Sidebar, Toolbar, Data Query).
5. Validate a11y (axe CI), ensure keyboard and focus order, and meet contrast targets.

**Compliance & Evidence**

* **Evidence Pack** attached to each release: build/test reports, CodeQL SARIF, secret-scan summary, SBOM + signature/attestation, OpenAPI + lint/diff results, monitoring snapshot, approvals. **Retention ≥ 1 year.**
* Map Security controls to ASVS/HIPAA where applicable; capture data flows and trust boundaries in the System Architecture doc.

**Risks & Mitigations**

* **Design/Code drift** → Design tokens from ThemeBuilder are the single source of truth; nightly visual regression (future).
* **Config sprawl** → All keys live in DB and are enumerated via /config/effective; changes audited.
* **SSE reliability** → Heartbeats, retry guidance, and fallbacks documented; JSON endpoints remain canonical for non-stream work.
* **Access misconfiguration** → Use app roles bound to AAD groups; PR checklist includes verification steps.

**Milestones**

* **Sprint 0**: finalize documents, OpenAPI 3.1 skeleton, DB SP contracts, CI skeletons, Figma Make brief.
* **Sprint 1**: Azure SSO login → Dashboard, /healthz /ready wired, basic metrics.
* **Sprint 2**: Config/Flags UI + SPs; /config/effective; audit log.
* **Sprint 3**: Jobs launch + SSE streaming; error taxonomy; rate limits.
* **Sprint 4**: A11y hardening, perf SLOs, evidence automation; RTM validation.
* **Release**: Promote to Prod with evidence pack.

**Acceptance Criteria (Sprint 0)**

* Vision, FR/NFR, Context, OpenAPI header, DB SP list, CI/CD plan, Figma→Kendo SOP drafted and in repo.
* Env URLs (or TBDs) captured; AAD app registration parameters listed.
* Evidence pack index added to /docs/12\_evidence\_pack.md.